TAMK University of Applied Sciences
Department of Machine and Manufacturing
Aeronautics
Mika Peltokorpi

Final thesis

Feasibility Study of Reliability Centered Maintenance Process

Applying RCM II approach to customer feedback in SW development environment

Supervisor	Head of Aeronautical Studies, Heikki Aalto
Commissioned by	Nokia Oyj, Jari Ekholm

TAMK University of Applied Sciences
Department of Machine and Manufacturing, Aeronautics

Author(s)	Mika Peltokorpi
Name of the report	Feasibility Study of Reliability Centered Maintenance Process
Number of pages	43
Graduation time	05.08.2009
Thesis supervisor	Heikki Aalto
Commissioned by	Nokia Oyj, Jari Ekholm

ABSTRACT

This thesis was a feasibility study of Reliability Centered Maintenance (RCM) method usage in customer feedback in SW development environment. In particular the version MSG-3 (contemporary version for aviation) or RCM II (contemporary version for power plants design) was used as reference method.

In customer interface of Nokia Devices there is a data base system used for customer feedback and analysis. The system meets very well its purposes, but it is not adopted for FMEA analysis. In Maintenance for Aviation course MSG-3 method was introduced. Because of it MSG-3 (RCM II) was taken as reference FMEA analysis method to this thesis.

The basic method was to run trough a RCM II style analysis for customer feedback report template and to analyze, how the existing data base could be modified to create RCM II style reports and analysis. For existing data base also CSA analysis was made in order to understand, if there is gaps vs. RCM II requirements and what kind of gaps there is (if any). mySQL language was used to describe basic data base structure of a RCM II compliant data base.

This thesis showed, that it is possible to calculate MTBF for SW in higher level even it is not possible in SW function level. RCM II approach had to be used in order to understand how deep functionality split can to be done, but still not losing possibility to calculate MTBF.

This thesis is good guide to persons unfamiliar to RCM and its history. This thesis also explains the basic requirements for data base design, if RCM compliant data base should be created. Most importantly silent information in customer interface becomes public domain

Analysis of customer has been already done with one application for a single industry partner feedback. This will be extended to other applications for this customer. If it is decided to implement RCM II style analysis to all customer feedback, the current data base should be modified accordingly. Also training material should be created and sessions should be arranged.

Keywords Final thesis, RCM, MSG-3, CSA, customer relations

Foreword

Impulse for this thesis was Maintenance for Aeronautical Systems course held by Juha Rintala. MSG-3 just seemed to fit for improving customer reporting interface performance trough very robust analysis method. Thank you for Juha for excellent lectures.

I thank Director of Degree Programme, Heikki Aalto, my solid line manager, Jari Ekholm, and my account team manager, Juha Luoto, for interesting graduation thesis. I thank also my former solid line manager, Vesa Parviainen, for original grant to study while working.

I thank my wife Ellen and my children supporting me with my studies.

Tampere August 2009

Mika Peltokorpi

TAMK University of Applied Sciences
Department of Machine and Manufacturing
Aeronautics

Table of Contents

1 Introduction....7

2 Reliability Centered Maintenance....9

2.1 History of Reliability Centered Maintenance....9
2.2 Functions....12
2.3 Functional Failures....13
2.4 Failure effects....13
2.5 Failure Modes and Effects Analysis....14
2.5.1 System....14
2.5.2 Analysis....17
2.6 Failure Consequences....18
2.7 Maintenance Tasks....20
2.7.1 Maintenance Task scheduling....22
2.7.2 Maintenance task list report for known root causes....23

3 CSA of RCM Process Implementation to customer feedback....24

3.1 Issue....25
3.1.1 Current state: Issue....25
3.1.2 Change Requirements: Issue....25
3.2 Test Case....26
3.2.1 Current state: Test Case....26
3.2.2 Change Requirements: Test Case....26
3.3 MTBF....27
3.3.1 Current state: MTBF....27
3.3.2 Change Requirements: MTBF....27
3.4 FMEA....28
3.4.1 Current state: FMEA....28
3.4.2 Change Requirements: FMEA....28
3.5 Task....29
3.5.1 Current state: Task....29
3.5.2 Change Requirements: Task....29

4 mySQL commands for creating RCM compliant database....30

4.1 Creating database....30
4.2 Creating tables....30

5 Recommendations of statistical analyses to use for incidence report data....33

5.1 Pareto analysis....33
5.2 MTBF....34
5.2.1 Example for MTBF calculation for a SW module....36
5.2.2 Comparable MTBF related values....38

6 Findings 39

7 Benefits of implementation 41

8 Further development 42

9 List of references 43

Appendices ....

Appendix 1: Incidence report field examples ....
Appendix 2: Customer communications package template ....

List of abbreviations

API	Application Programming Interface
CAA	(UK) Civil Aviation Agency (earlier Civil Airworthiness Agency)
CSA	Current State Analysis
DoD	(US) Department of Defense
DT	Down time
EBA	Evidence Based Acceptance
FAA	Federal Aviation Agency
F(M)E(C)A	Failure (Mode,) Effect (and Criticality) Analysis
MSI	Maintenance Significant Item, maintenance task
MSG-3	Maintenance Steering Group revision 3 (RMC for Aviation)
MTBF	Mean Time Between Failures
RCM	Reliability Centered Maintenance
SW	Software
TBF	Time Between Failures
TCP	Transmission Control Protocol
TTF	Time To Fix
TTV	Time To Verify
UDP	User Datagram Protocol
UT	Up Time
WAS	Works As Specified

1 Introduction

This thesis is inspired by the most common misconception about SW development, which is described in following phrase:

"If it is fixed once, it will fail never again" (unknown SW developer 2008)

This misconception will end up to the conclusion, that if a failure can be fixed for good every time it is detected, it can not be failed again. Thus statistical measurements, such as Mean Time Between Failures (MTBF), can not effectively be measured to the SW. In a source code level that applies very well, but if you consider the same problem in higher level, the truth can be totally different.

The SW module or component can be failing by multiple reasons. The failure could for example be related to missing component or missing upgrade of a component. Or the failure could occur due some interference from other SW component or module communicating with the SW component or module demonstrating ill behavior. What the end user then sees is a missing, limited or otherwise bad functionality of a device or application, albeit the reason for this failure will usually remain ambiguous. Even to the organization creating the product it is often hard to find out the real root cause to the failure without proper analysis methods. In world of maintenance similar need of analysis tools evolved to Reliability Centered Maintenance approach described by John Moubray in his book (John Moubray 1997).

Nokia customer interface team is responsible of handling customer feedback during product development phase. In the past there has been some focused analysis for certain purposes in the team; most notably Works As Specified (WAS) reports reduction and Evidence Based Acceptance (EBA) exercises last year. Even the root causes were found and the main target of reducing amount of customer feedback the main problem of those analyses were, that those were not conducted in a systematic manner. The only re-usable part of the end reports of those analyses are the root causes for failures of certain test cases. The relation between the reason for the failure (e.g. root cause) and the circumstance (e.g. failure mode) the failure happened was not defined on those. The business impact or other effect of the failure was not defined, either. In RCM process all of these will be defined. Therefore, as the end result of analysis, RCM gives the list of root causes of the failures having the most business critical impact. As customer interface team is a part of customer interface, its operation is strongly driven by the factor of business impact a tool such as RCM would greatly help to focus resources in the most economically feasible way. And that kind of tool this team has been lacking.

The target of this thesis is to evaluate how RCM process could be utilized in SW development environment. In order to achieve this customer program has to be subjected to product development methods. In the scope of this thesis customer program is considered to be equipment having different kind of modules and entities performing certain functionalities. Each of the customer reports will be considered as failure of customer program to achieve adequate performance.

Out of the scope of this final thesis the Failure Mode and Effects Analysis (FMEA-analysis) will be done for customer feedback of single industry partner and to a single application and five products in a same product family. Further outlining of source data has been done to select the customer reports related to the main testing tool they are using for this feature, as those customer reports have real unified test case behind those. This amount of data to be analyzed is statistically valid, but not too large for final thesis purposes. There are some references to the test results in this thesis, but detailed results are not revealed due confidentiality reasons.

2 Reliability Centered Maintenance

RCM is related to Total Quality Management (TQM) approach, but due holistic approach to quality, TQM is not giving specific instructions how to implement quality management in a specific task. TQM is also concentrating of quality management in organizational or process level, not in engineering or product development level. However, the core idea of TQM, quality management being continuous process, is very well adopted in RCM:

> *This emphasis on what the asset does rather than what it provides a whole new way of defining the objectives of maintenance for any asset – one of which focuses on what the user wants. This is the most important single feature of RCM process, and is why many people regard RCM as 'TQM applied to physical assets'. (Moubray 1997, p. 21)*

Contemporary RCM methods are optimized to be used on maintenance of complex systems such as power plants (RCM II). MSG-3 is de facto RCM standard of managing maintenance in aircraft industry.

2.1 History of Reliability Centered Maintenance

Until the late 1950 Federal Aviation Agency (FAA), US government's regulatory office for airline maintenance had been more and more concerned on failure rates on certain engine types. In order to solve these reliability problems, it established FAA/Industry Reliability Program. Term Reliability-centered maintenance was mentioned first time was introduced in the report made by Nowlan and Heap. The first versions of the decision making diagrams were published in 1968. (Järviö 2004, 2-4).

In late 70's United Airlines was required to report how it will develop its maintenance program. That program was funded by US Department of Defense (DoD) and the end report was called "Reliability-centered Maintenance" (Nowlan, Stanley, Heap, 1978). Contemporarily MSG-1 had evolved to more sophisticated MSG-2. MSG-2 still had bottom-up approach for analysis, but adoption of RCM approach turned the FEA analysis process to top-down approach in MSG-3. MSG-2 is still applied to maintenance of military and civil aircrafts such as L-1011 and DC-10. (Friend 1992, 48-50), (NAVAIR 1998, 13-15).

Until the 1980's US aviation industry and military had been the main drivers to reliability centered approach in the maintenance. But the interest of TQM and RCM ideology especially in mining and manufacturing sectors, led application of RCM process in maintenance departments of companies in the other industry sectors. That led of creation of RCM II. Today SAE JA1011 defines the minimum requirements for RCM process and also US military is nowadays following the SAE JA1011 basic principles in it's internal requirements such as NAVAIR 00-25-403 (Revision 1). (Moubray 1997, 321-326)

As RCM and MSG are nowadays in process mind the same, further on in this thesis RCM refers to RCM II / MSG-3.

History of RCM and related specifications is described in Figure 1.

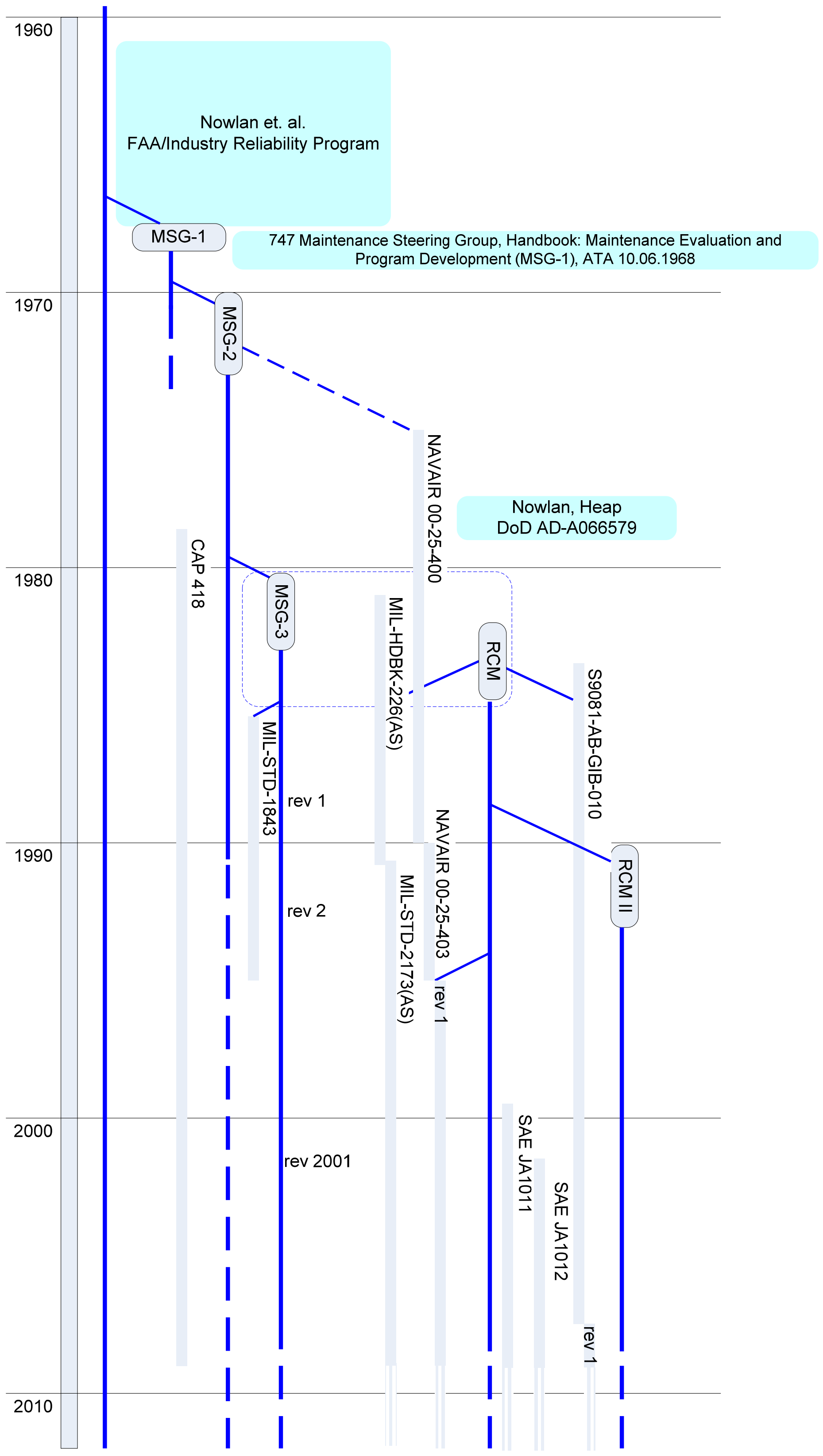


Figure 1: History of Reliability Centered Maintenance

2.2 Functions

In RCM methodology *function* is described as state, where system or component is *working within its design limits*. In SW development environment running a test case with positive (passed) verdict can be considered as such *function.* Customer interface is working within its design limits, when there is no customer *incidence reports* inflow – or it is minimal. Figure 2 shows a template of RCM compliant incidence report in data base system.

Customer input			
Date	Title		Report ID
Area	Test Case		Customer
Product SW version:		Product platform SW version(s):	
Incidence body (description of problem):			
FMEA			
The list of reports having same Test Case:			
Failure:	Failure Mode 1:	Failure Mode 2:	Failure Mode 3:
The list of reports having same root cause (Failure Mode):			
Failure consquence			
Hidden (y/n):	Safety (y/n):	Operational (y/n):	Type:
Failure Effects:			
Redesign (y/n):	Integration (y/n):	Operational (y/n):	
Configuration (y/n):	Sales (y/n):	Safety (y/n):	
Tasks			
Maintenance Task 1			
…			

Figure 2: Template of a customer incidence report.

2.3 Functional Failures

In RCM *functional failure* is described as a state, when a function is not operating within its design limits. Thus also outperforming specified performance level would be a failure, which is traditionally seen as asset, not deficiency. Analogy to customer reporting for incident report with outperforming the specification limits would be a situation, where a test case is passed even verdict should be failed.

In the first analysis done to real data using this thesis as basis the *functional failures* had following categories (N=550):

- Test Case can not be run (~1%)
- Test Case can be partially run (~1%)
- Test Case is passed partially
- Test Case is not passed (~98%)
- Test Case passed, but should fail

See Appendix I for hypothetical examples of the functional failures. Percentages above describe hit rates for different failure categories identified for the *incidence reports* after FMEA analysis was done to real data.

2.4 Failure effects

Each of functional failure should have an *incident report* created for it. In RCM the *incidence report* is considered always as implication of functional failure of the system that is under RCM analysis. Thus, when RCM principles are implemented in customer interface, each customer *incidence report* is considered as implication of *functional failure* for customer interface team. If no product specific functional failure is found during FMEA, the *failure effect* is to communicate customer about it. In RCM perspective main target of the customer interface team should be minimizing the inflow of incidence reports, as those have always economical impact to the company. This idea is the most important hypothesis on this thesis.

The list of *failure effects* categories created during this thesis work for FMEA analysis are:

- Communication (least severe)
- Test Case redesign
- SW Configuration
- SW Integration
- SW Redesign
- Sales blocking
- Safety (most severe)

Single report can have multiple failure effects. For example SW needs redesign and that has to be communicated to the customer. Failure effect is explained in detail in *maintenance task*. Maintenance tasks and failure effect selection are described in Chapter 2.7.

2.5 Failure Modes and Effects Analysis

2.5.1 System

SW is usually described with ISO/OSI model that consists of seven vertical layers. In the Figure 3, a theoretical module is illustrated. In this model there is three applications two of which have access to the same transport layer (e.g. TCP), and third to other one (e.g. UDP). Communication between layers is created by Application Programming Interfaces (APIs). Within each independent layer module there can be many sub functions, each of which has several APIs.

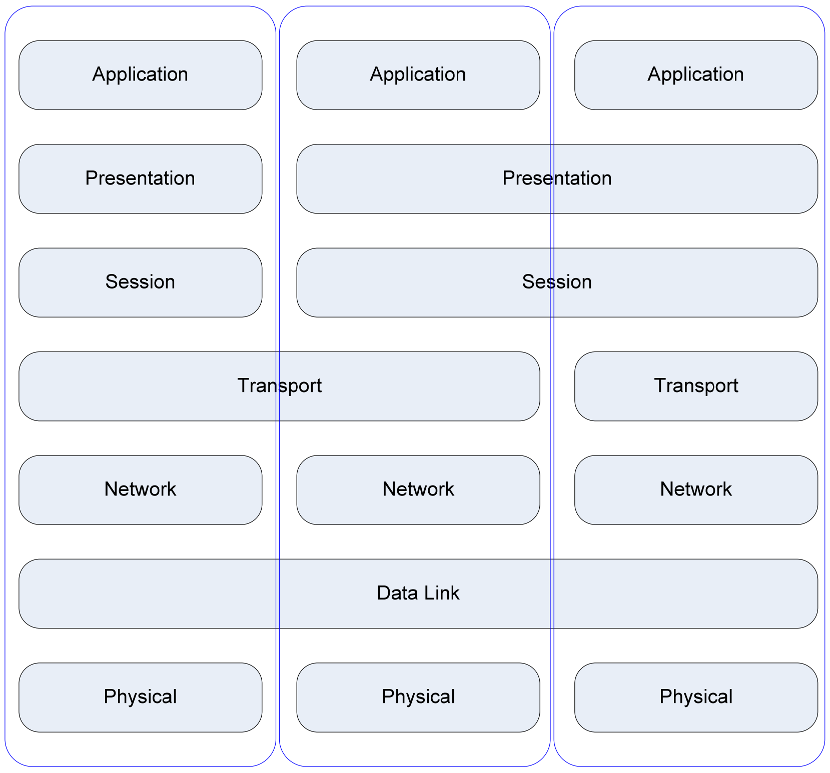
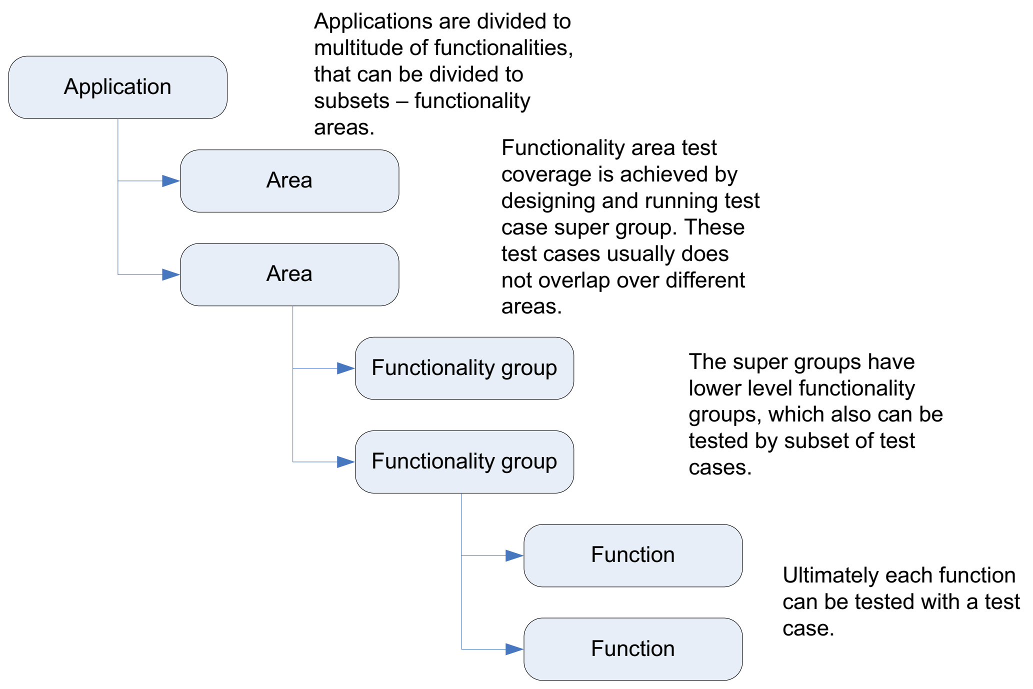


Figure 3: ISO/OSI model

Each application usually has dedicated specialists coding those, which makes it more feasible to make FMEA study to each application separately. Also product level test cases are usually structured so, that division is done according applications and sub functionalities of those (see Figure 4). So all of this supports starting functionality split in vertical scope in stead of horizontally in ISO/OSI model.



Applications are divided to multitude of functionalities, that can be divided to subsets – functionality areas.

Functionality area test coverage is achieved by designing and running test case super group. These test cases usually does not overlap over different areas.

The super groups have lower level functionality groups, which also can be tested by subset of test cases.

Ultimately each function can be tested with a test case.

Figure 4: Application functionalities and test cases.

SW function should not be considered as *function* understood in RCM. SW function can have multiple independent *functions*. SW function has inputs, atomic code and non-atomic code, which comprises of multiple *functions* (see Figure 5). Each of which should be tested separately. Naturally the customer feedback is not in SW function level, but it is generated by test results on the customer's test case set. However, when code level analysis it should be understood which kind of *functions* are in the code under analysis and how to design proper test coverage of these *functions*. Code level analysis will be done by R&D.

```
function (input1, input2, ...)
         {
         interrupts.disable()

                  atomic code

         interrupts.enable()

                  non atomic code

         interrupts.disable()

                  atomicity for clean
                  exit, kernel will
                  enable interrupts
         };
```

SW function has certain inputs, that has to be tested with several test cases. (multiple functions).

Atomic code is easy to test, as it will be never leaved until end of atomic code. (single function)

Non-atomic code should not have different behavior, if it's execution is interrupted. In loading situations that may differ (multiple functions)

Figure 5: SW function vs. RCM functions

2.5.2 Analysis

FMEA analysis should always be done by competent body. In case of customer incidence reports, the main responsibility of running FMEA analysis will be on the customer interface organization assigned for the task. This activity will be supported by the stake holders in R&D and customer.

Figure 6 illustrates, what initial approach to functions, functional failures and failure modes was taken in this thesis. As this was the 1st time RCM style approach was taken to the customer incidence reports, the idea was to collect as many as possible combinations there is available. That means of collecting also (seemingly) impossible choices to each category. As the end result of the first analysis work done using this thesis Failure category 5 (see Figure 6) was excluded as it should never occur in customer feedback. Also some second level Failure Modes could be combined during these iteration rounds for that data.

Function	Failures	Failure Modes		
Test Case run	**1** Test Case can not be run	**A**	Server	**1** Configuration
	2 Test Case can be partially run			**2** Malfunction
	3 Test Case is passed paritally			**3** Funtionality Interference
	4 Test Case is not passed	**B**	Test Case	**1** Test Case Direct Flaw
	5 Test Case passed, but should fail			**2** Test Case Content Not Supported
				3 Test Case Instruction Not OK
				4 Test Case Indirect Flaw
		C	Human	**1** Test Setup
				2 Test Case Instruction Not Understood
				3 Device Capabilities Not Understood
				4 Device Capabilities Not Available (early SW)
				5 Wrong Specification Used
				6 Wrong Test Case Verdict
		D	Integration	**1** Component Missing - Variant
				2 Component Fail - Variant
				3 Component Configuration - Variant
				4 Component Upgrade not done - Variant
				5 Component Missing - Core
				6 Component Fail - Core
				7 Component Upgrade not done - Core
				8 Component Configuration - Core
		E	Support	**1** Deprechiated feature
				2 Higher or adjasent module feature
				3 New feature request (not suppored)
				4 Partial support only

Figure 6: Initial functions, functional failures and failure modes.

In addition to Figure 6, the analysis is feasible to be started from test case one to the last test case. And for each incidence report *third level* Failure Mode has to be defined. For the third level Failure Mode, it has to be checked, if it is existing failure mode or new failure mode for the test case incidence report was reported to. This level of detail is still effective, but not too detailed for customer feedback in SW development environment. But in some other implementations even more detailed failure mode division may be needed. See Appendix I for example of Failure Modes. After defining failure mode and failure effect for each incidence report, the *failure consequence* has to be defined. That is described in next chapter.

The end result of a FMEA analysis of a customer incidence report is root cause for that report. If the root cause is in platform SW level the root cause combining can – and should be – done for those root causes also across different applications.

2.6 Failure Consequences

In RCM failure consequences are categorized by four variables:

- Perceptivity (hidden/detectable)
- Safety
- Operational
- Economical

The MSI (Maintenance Significant Item) type – or Failure Consequence category - is selected by logic described in Figure 7.

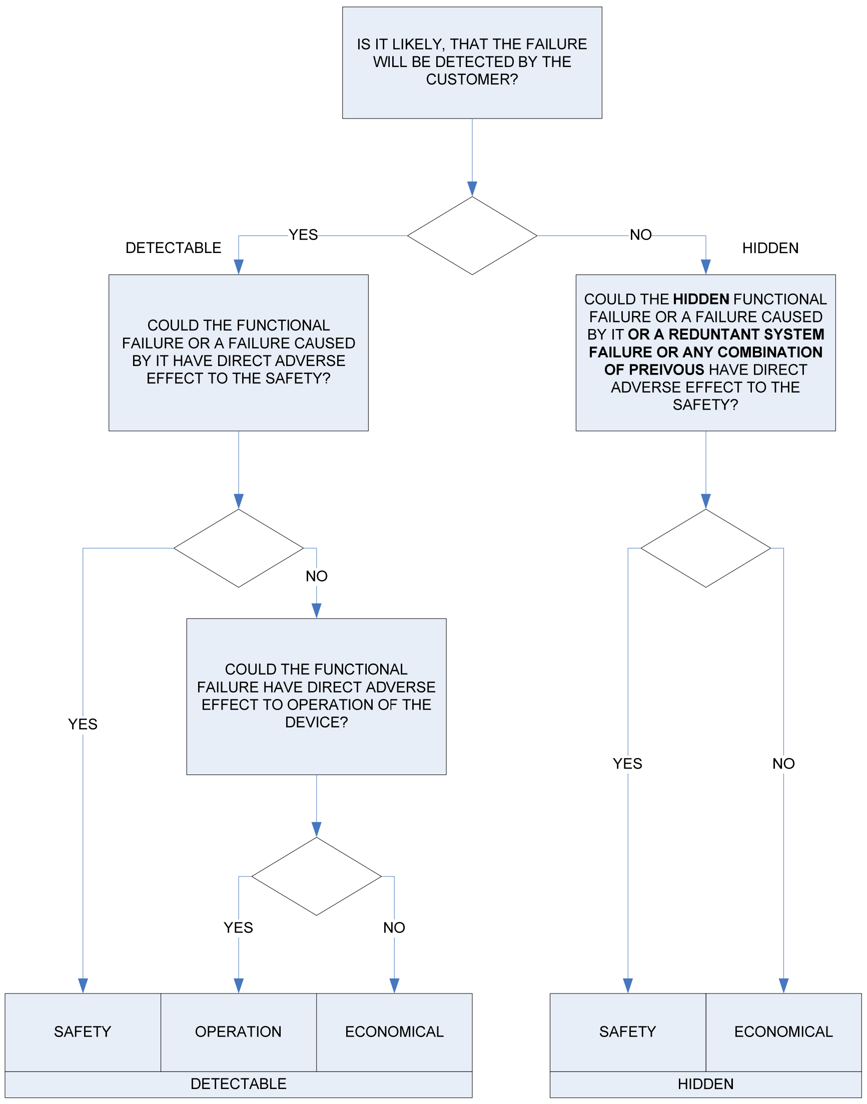

IS IT LIKELY, THAT THE FAILURE WILL BE DETECTED BY THE CUSTOMER?
YES
DETECTABLE
NO
HIDDEN
COULD THE FUNCTIONAL FAILURE OR A FAILURE CAUSED BY IT HAVE DIRECT ADVERSE EFFECT TO THE SAFETY?
COULD THE HIDDEN FUNCTIONAL FAILURE OR A FAILURE CAUSED BY IT OR A REDUNTANT SYSTEM FAILURE OR ANY COMBINATION OF PREIVOUS HAVE DIRECT ADVERSE EFFECT TO THE SAFETY?
YES
NO
COULD THE FUNCTIONAL FAILURE HAVE DIRECT ADVERSE EFFECT TO OPERATION OF THE DEVICE?
YES
NO
YES
NO
SAFETY
OPERATION
ECONOMICAL
DETECTABLE
SAFETY
ECONOMICAL
HIDDEN

Figure 7: RCM logic tree

In RCM the fixing priority for the MSIs across to the MSI categories is

- Hidden Safety
- Detectable Safety
- Detectable Operation
- Hidden Economical
- Detectable Economical

In customer interface, if there is no other impact from *an incidence report*, there is time and money spent for analyzing it. In such situation the *failure consequence* for that incidence report would be *detectable economical*.

2.7 Maintenance Tasks

Each failure mode identified by FMEA analysis should have at least one *maintenance task* (or *maintenance significant item*, MSI) defined to it. According to RCM definitions the maintenance task should always be both *effective* and *economically feasible* to be implemented. Good customer service requires response to the customer, so there can not be total exclusion of all maintenance tasks for *a root cause*, which is allowed in for example in MSG-3 used in aviation industry. Therefore at least communication task will always be defined for every *root cause*.

In case of customer interface incidence reports *effective* maintenance task should include at least:

- Root cause description
- List of the test cases (functions) failing because of the same root cause
- Estimated time to have the root cause fixed, if the functions (test cases) should be supported according the design specification of the product
- Inform customer, what is required from it if it wants the root cause preventing functions to be fixed (test cases to pass), if the design specification of the product indicates, that this function is not supported by it
- External reason(s), why functional failure occurs, if such is known. This kind of failure could be for example coding error in the test case.

See Appendix II for example of maintenance task list containing MSIs.

After effective maintenance task(s) have been defined, *economical feasibility* could limit the implementation for more severe maintenance tasks, than a communication tasks. Figure 8 describes that selection procedure. For each root cause either a communication task or a higher level maintenance task complemented with a communication maintenance task will be created.

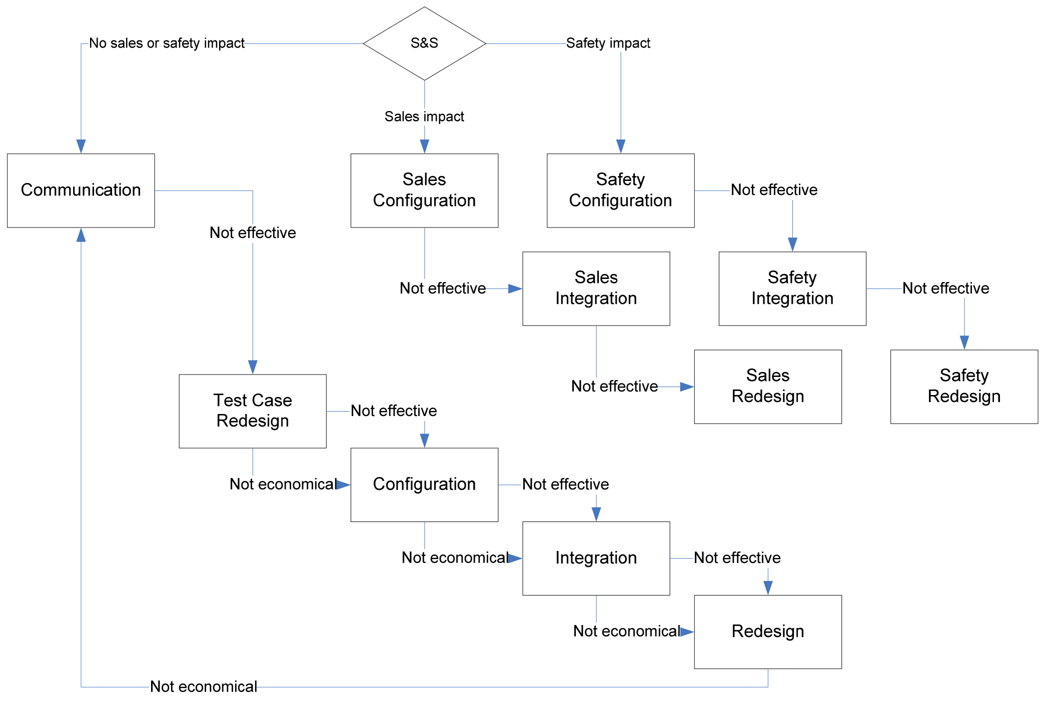


Figure 8: Maintenance task class (failure effect category) selection

Redesign is always effective task, but it may not be economically feasible. That kind of situation can occur for example:

- If the redesign requirement would impose large amount of coding effort and testing requirements to the SW and the functional benefits of those changes would be minimal.
- If the redesign requirement imposed to a SW in end of its life cycle and major changes to the code would be needed.

In these cases the maintenance task would be communication to the customer that the SW would need redesigned, but it would not be economically feasible to redesign that feature.

Safety related and *sales blocking* (S&S in Figure 8) tasks are always considered as *economically feasible*. After analysis has been done it might be, that these tasks can be configuration, integration or configuration tasks for R&D, but for customer interface those are still *sales blocking* or *safety related* tasks.

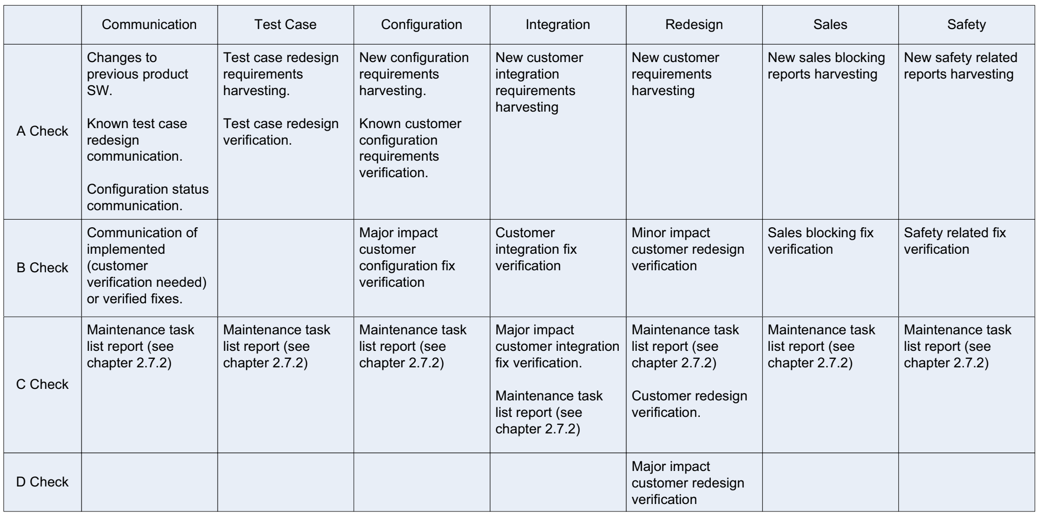
2.7.1 Maintenance Task scheduling

In aviation industry maintenance tasks are scheduled by MTBF, *failure consequence* and *failure effect*. MTBF is taken into account, as mechanical wearing of a system, module or component can be depended by flight hours or number of flights. The schedule could be defined for example with following intervals:

- A Check, 500 flight hours
- B Check, 3 months
- C Check 12 months
- D Check 4-5 years

In customer interface maintenance task scheduling can be made for example in following way:

- A Check
  - Every SW delivery to customer
- B Check
  - Every product sales SW delivery to customer
- C Check
  - Customer testing start for a new product
- D Check
  - Platform SW redesign delivery to customer testing



	Communication	Test Case	Configuration	Integration	Redesign	Sales	Safety
A Check	Changes to previous product SW. Known test case redesign communication. Configuration status communication.	Test case redesign requirements harvesting. Test case redesign verification.	New configuration requirements harvesting. Known customer configuration requirements verification.	New customer integration requirements harvesting	New customer requirements harvesting	New sales blocking reports harvesting	New safety related reports harvesting
B Check	Communication of implemented (customer verification needed) or verified fixes.		Major impact customer configuration fix verification	Customer integration fix verification	Minor impact customer redesign verification	Sales blocking fix verification	Safety related fix verification
C Check	Maintenance task list report (see chapter 2.7.2)	Maintenance task list report (see chapter 2.7.2)	Maintenance task list report (see chapter 2.7.2)	Major impact customer integration fix verification. Maintenance task list report (see chapter 2.7.2)	Maintenance task list report (see chapter 2.7.2) Customer redesign verification.	Maintenance task list report (see chapter 2.7.2)	Maintenance task list report (see chapter 2.7.2)
D Check					Major impact customer redesign verification		

Figure 9: Maintenance task scheduling

2.7.2 Maintenance task list report for known root causes

Customer interface's main task is to minimize incidence report amount. Thus maintenance task list for known root causes should be created. This kind of document should be organized so, that it has maximum impact on reduction of the amount of incidence reports. This chapter describes basic structure of this document and Appendix II is an example template of it.

First it is good to list known functional changes (redesigns, fixes and depreciated features) in upcoming version of this part of product SW. This should eliminate incidence reports that would be created because customer is not aware of changes, when new version is deployed.

Then the tasks are organized in following order:

- *root causes*, which *failure effect* is *safety related* or are *sales blocking*
    - Note: this chapter is omitted from Appendix II
- If MTBF is collected for root causes, r*oot causes* with frequent MTBF interval and that have *failure effect(s)* higher than communication
    - threshold frequency for root causes on this category should be defined
        - for example MTBF <30 days
        - or MTBF = 0; every SW sent to testing has this root cause reported on it because a test case has/had a flaw
    - this may be feasible for configuration or integration class tasks
    - Note: this chapter is omitted from Appendix II
- the rest of *root causes*
    - Highest level sorting of *root causes* by *failure effect* from Communication tasks to Redesign tasks (ascending)
        - Second level sorting of *root causes* by *failure consequence* from Detectable Operational to Detectable Economical (descending)
            - Third level sorting of *root causes* by *number of incidence reports* (descending)

3 CSA of RCM Process Implementation to customer feedback

This chapter focuses on the changes needed to current reporting system, if RCM methods would be implemented in the database used by customer interface team. Appendix I has examples of each part of data base structure described in this chapter.

In the information transfer wise customer's data base is in upstream and program data base is in down stream. In middle of those is the database used by customer interface group. As the RCM analysis should be done by customer interface, the R&D system does not need changes for RCM purposes. If the RCM type analysis would be implemented in R&D system, the below CSA analysis should be done also to it.

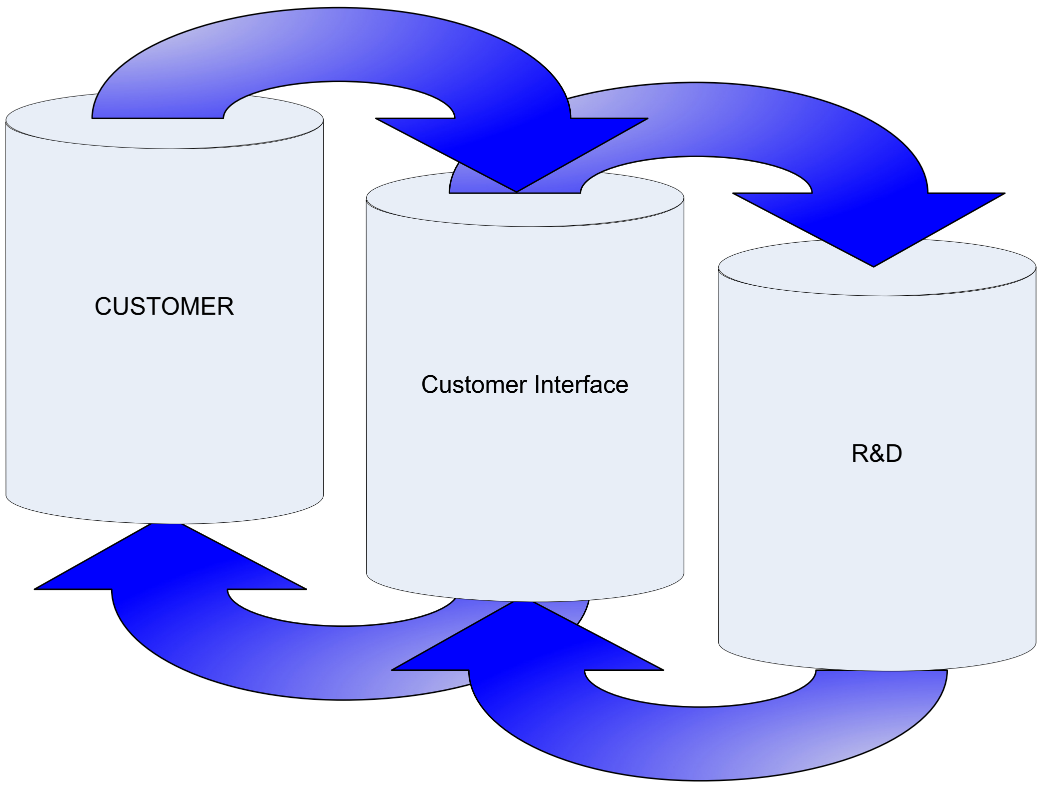


Figure 10: Customer interface dataflow

Connection between customer data base and customer interface data base can be real data base connection or manual transfer. For simplicity reasons further chapters it is assumed to be real connection. If there is no real data base connection to customer's incidence report data base, in next chapters "customer" is referring to person responsible to transfer customer feedback (contact by e-mail, telephone, spread sheet or word processor document) manually to customer interface data base.

3.1 Issue

3.1.1 Current state: Issue

Customer

- Data delivered to customer interface data base consists: product and product SW, sequential issue number for the customer, issue description in plain text format, criticality level of the issue for the customer.
- Customer can redefine incidence report status between various open/inspection on-going or closed/investigation stopped statuses.

customer interface

- customer interface system amends the *incidence report* with information about issue criticality level for customer interface.
- customer interface can define new (internal) incidence report status and that status is updated to the customer data base.
- There is possibility to transfer *incidence report* to R&D for further inspection.
- All of issue data can be transferred to the analysis tool.

3.1.2 Change Requirements: Issue

Customer

- Currently the level the *incidence reports* are delivered by the customer is sufficient - assuming that the incidence report itself contains needed information. Thus no modification is needed for RCM purposes.

customer interface

- No changes are needed for RCM purposes.

3.2 Test Case

3.2.1 Current state: Test Case

Customer

- The test case identification is not automatically implemented. Usually that information is embedded either in the title of the incidence report or in the body of it, but there is no fixed manner enforced, how it is delivered to supplier (customer interface system). Thus this information may be also missing from the incidence report.
- Test case description quality varies. It could also include link to customer test server. The customer server could be accessible by supplier or not. In latter case joint test session may be needed before passing the issue to R&D system.

customer interface

- customer interface has to request test case related information from the customer in order to be able to process the issue. For high priority reports the issue has sometimes to be forwarded to R&D before that is done.
- Issue title is separate data item that can be transferred to the analysis tool.

3.2.2 Change Requirements: Test Case

Customer

- Unique customer test case identification should be always available in each incidence report. Preferably in separate field, for example unique and uniform title is enough for this purpose.

customer interface

- Test case identification is very vital for RCM level MTBF calculation. Unique uniform title is enough for this purpose. This item should be transferrable to the analysis tool.
- customer interface should not accept incidence report, if it does not include test case identification and clear description how the test case can be run.

3.3 MTBF

3.3.1 Current state: MTBF

Customer

- Customer delivers information about issue detection time, SW version and product. Customer also reports, when the issue has been solved.
- Customer may reopen report, if there is new fail in the test case.
- Each state change is time-stamped.

customer interface

- The report status can be changed by customer interface. Each state change is time-stamped.
- customer interface team can communicate fix availability in comment field.
- Time stamp of creating an report in customer interface database and last handling time are delivered to the analysis tool.

3.3.2 Change Requirements: MTBF

Customer

- No need for changes in the data base.
- However, current process of reopening closed reports does not meet RCM requirements. Thus if incidence report has already been opened and closed for the product, new incidence report should be opened instead of reopening old one.

customer interface

- In order to fully meet the MTBF calculation requirements, the database should have possibility to derive SW sub module release versions and times from device SW version information.
- If MTBF data on the report would include also split between upstream and downstream times, fix availability time and fix verification time, those should have own separate item in the incidence report.

- Automatic calculation of processing time, time to fix availability and time to verify would ease MTBF-calculation, when those are sent to analysis tool. Fix availability time stamp can be implemented as manual entry by customer interface team.

3.4 FMEA

3.4.1 Current state: FMEA

Customer

- This has not been implemented at all.

customer interface

- This has not been implemented at all.

3.4.2 Change Requirements: FMEA

Customer

- No need for changes, FMEA analysis is done by customer interface.

customer interface

- Full range of RCM related items should be implemented to the incidence report. These items would include:
    - Failure (most commonly: test case not passed)
    - Failure mode (multiple level, of which last one free from text field)
    - Unique failure mode ID creation in RCM style
    - Failure effect analysis (Hidden, Safety, Operational, Economic) => Effect type
- All of items mentioned above should be transferrable to analysis tool. These items may be changed in the analysis tool and modified values should be able to be sent back to the customer interface system.
- Analysis tool itself should have templates for statistical analysis. The extracted incidence data should be easy to be filtered to separate analyses according to for example incidence report priority or root cause.

3.5 Task

3.5.1 Current state: Task

Customer

- This has not been implemented at all.

customer interface

- customer interface group can communicate the task via comment field.

3.5.2 Change Requirements: Task

Customer

- No need for changes, RCM analysis is done by customer interface.

customer interface

- The following entry fields should be amended to the incidence report:
    - Task type (Communication, Configuration, Test Case Redesign, Integration, Product Redesign).
    - Link to maintenance task and task ID Current system would allow short description and link implementation within incidence report in a convenient way. This may be quite an easy to implement automatically, as the current data base system supports it.

4 mySQL commands for creating RCM compliant database

This chapter describes one way of creating a database that meets minimum requirements to create a database to handle RCM style incidence reports in SW development environment. As I have some knowledge about SQL, mySQL style definition is used to define the data base structure in this chapter.

The presented split between `report, sw, test_case` and `fix` tables is optimized for minimizing duplicate data in this relation database. Some of needed interdependencies between these tables are described later. In addition to these tables, reference tables for failures and failure modes may be needed. Those will contain plaintext information about the failure/mode corresponding to certain ID.

The functionality for the tables can be done in web interface using for example PERL or PHP scripting language. Implementation of that is out of scope of this graduation thesis and can be considered as one of best further development item for this graduation thesis. These tables are to be considered as starting point for specifying implementation specific RCM compliant data structures, only.

4.1 Creating database

```
CREATE msg3
```

4.2 Creating tables

The active database has to be selected with

```
USE msg3
```

Figure 11 shows, how to create incidence report table. `Root_cause_id` is needed for statistical analysis purposes described in Chapter 5.2. In this case additional table for failure modes (e.g. `modes`) should be created and it should include plain text information about the global list of failure modes in three separate levels. Maintenance tasks are in separate table.

CREATE TABLE report (	
report_id INT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,	Incidence report id
product VARCHAR(20),	Product under test
prod_sw VARCHAR(20),	Product SW
created DATE,	Incidence report creation date
test_case VARCHAR(20),	Failure
test_area VARCHAR(20),	Test case group
sw_module VARCHAR(10),	SW sub module (from FMEA)
fix_date DATE,	Fix date
fix_duration INT,	Fix duration (DT\\product)
fix_id VARCHAR(25),	Fix ID (MSI ID)
verification_date DATE,	Fix verification date
failure_id VARCHAR(40),	Failure ID
mode1_id SMALLINT UNSIGNED NOT NULL,	Level 1 MSG-3 type Mode ID
mode2_id CHAR(1),	Level 2 MSG-3 type Mode ID
mode3_id SMALLINT UNSIGNED NOT NULL,	Level 3 MSG-3 type Mode ID
root_cause_id VARCHAR(10),	Level 1..3 type Mode ID
hidden SMALLINT(1),	Effect hidden?
safety SMALLINT(1) ,	Effect safety?
operational SMALLINT(1),	Effect operational?
economical SMALLINT(1),	Effect economical?
task_type VARCHAR(10));	E.g. communication, component upgrade, redesign

Figure 11: Report table

Product SW version reference table (`sw`) is needed if SW sub module level MTBF calculations are planned to be done. In theoretical example in Figure each product SW contains two main SW modules. For MTBF calculations `sw` each table entry has to define also creation date for each SW sub module that is included to corresponding product SW. Theoretical structure for such `sw` table is described below.

CREATE TABLE sw (	
product VARCHAR(20),	Product
sw_ver VARCHAR(20),	Product SW version
sw_date DATE,	Product SW creation date
sub_SW_1 VARCHAR(20),	SW sub module 1 (plaintext)
sub_SW_1ver VARCHAR(10),	SW sub module 1 version in the product
sub_SW_1date DATE,	SW sub module 1 version creation date
sub_SW_2 VARCHAR(20),	SW sub module 2 (plaintext)
sub_SW_2ver VARCHAR(10),	SW sub module 2 version in the product
sub_SW_2date DATE);	SW sub module 2 version creation date

Figure 12: SW table

Separate table for test cases is created with following way (where `test_case.id` equals `report.test_case`). `test_case.url` is plain text field corresponding to web address containing details about that test case.

```
CREATE TABLE test_case (
            id VARCHAR(20),
            title VARCHAR(40),
            url VARCHAR(255));
```

Separate table for maintenance tasks (where `fix.id` equals `report.task`). `test_case.url` is plain text field corresponding to web address containing details about that maintenance task.

```
CREATE TABLE fix (
           id VARCHAR(20),
           category VARCHAR(40),
           url VARCHAR(255));
```

In this chapter both `test_case` and `fix` descriptions are behind URL in order to give flexibility on the way the test case or maintenance task is described. The URL can point to actual test case execution web page or other web page, word processing document, spread sheet document or an external database containing the test case description.

5 Recommendations of statistical analyses to use for incidence report data

5.1 Pareto analysis

Pareto Diagram gives information, how root causes will contribute to over all incidence report amount. As the different products, SW sub modules and platform SW generations will have different amount of issues, scaling to 100% maximum instead of absolute numerical value gives more comparable value across those. Therefore 100% scaled x-axis and y-axis is better, than using the absolute values.

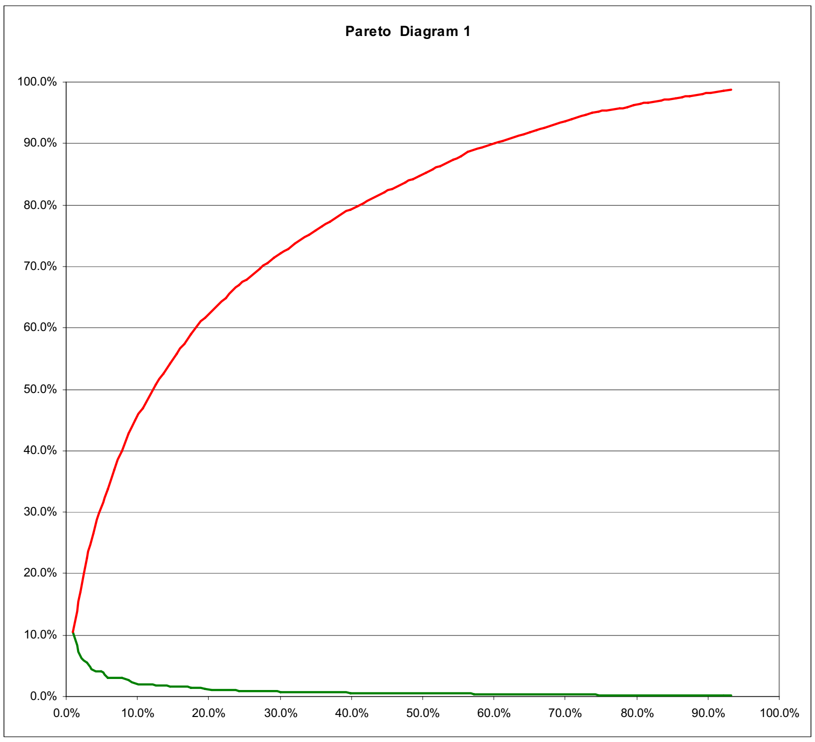


Figure 13: Pareto Diagram of test results.

In Figure 13 green line describes relative amount of incidence reports caused by a single identified root cause. The root cause hit data is sorted from greatest hit number to lowest. Red line indicates cumulative number for the same data. With this statistical tool it is possible to estimate effectiveness of RCM process. From the Pareto diagram, following formula can be created:

$$P(\sum x_{\%}, y_{\%} = 100\%) = \frac{y_{\%}}{x_{\%}} - 1$$

In Figure 13 this reference value is:

$$P(\sum x_{\%}, y_{\%} = 100\%) = \frac{72\%}{28\%} - 1 = 1{,}57$$

This indicates that significant benefits can be achieved easily with RCM process.

When creating the Pareto Diagram, the root causes are also sorted to descending order by hit number of the root cause. That ordered list can be considered as initial priority handling list for the customer incidence reports.

5.2 MTBF

For Mean Time Between Failures (MTBF) calculations to any system it is crucial to understand, when the system has a fail and when that fail is considered to be fixed in the evaluation context. This chapter focuses on how to calculate MTBF related data for customer incidence reports in different contexts.

Time Between Failures (TBF) should be calculated to all *root causes* (instead of *test cases*) at least per product SW. As described in chapter 3.3 each *new failure* should have new incidence report in order to get proper MTBF calculation. MTBF for a root cause is a mean value of all TBF:s for that root cause in that product. Thus n in next formula is determined by the amount of incidence reports for that product with the same root cause.

$$MTBF = \sum_{i=1}^{n} \frac{TBF_i}{n}$$

In order to measure TBF times, each customer incidence report has to have field for creation date and also for the time, customer reports the problem solved. Now, if the same *root cause* has new incidence report a TBF can be defined.

Figure 14 describes basic principle how to calculate MTBF for customer incidence reports. In the figure below *root cause is the same for both incidence reports.* The test case may be the same for both incidence reports, but can also be different. The fix task should be different for these two incidence reports, as the root cause is same and there was working component fix earlier for that root cause.

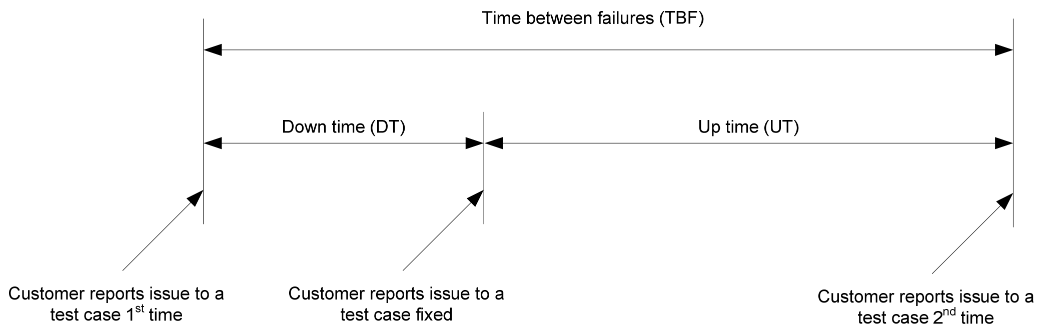

Time between failures (TBF)
Down time (DT)
Up time (UT)
Customer reports issue to a test case 1st time
Customer reports issue to a test case fixed
Customer reports issue to a test case 2nd time

Figure 14: TBF calculation principle.

Down Time (DT) can be split to two components, Time To Fix (TTF) and Time To Verify (TTV), in order to get better quality metrics for customer issue handling process. TTF and TTV gives the handling times for incidence report in downstream and upstream data flow respectively as described in Figure 10 in Chapter 3. Or time to analyze and implement the fix (TTF) and time to verify the fix (TTV).

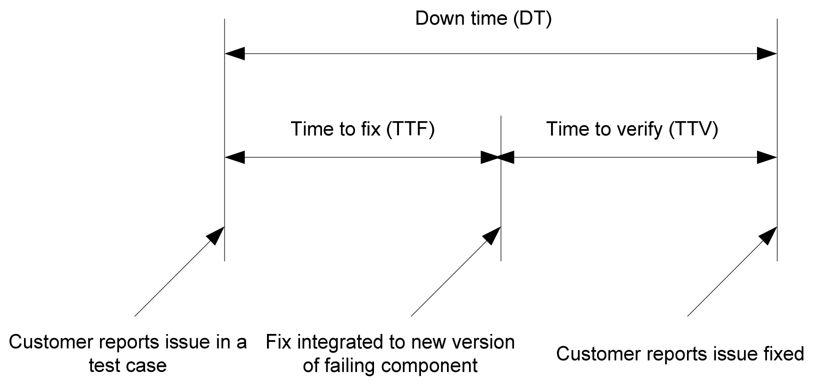

Down time (DT)
Time to fix (TTF)
Time to verify (TTV)
Customer reports issue in a test case
Fix integrated to new version of failing component
Customer reports issue fixed

Figure 15: Down time split.

TBF could be calculated also over a SW sub component or test area. In case of SW sub component case the down time (DT) time to fix (TTF) time starts from creation date of failing SW module. TTF calculation stops, when 1st product (any of the products) has the (later to be proven to work) fix; time to verify (TTV) time calculation starts at the same point. TTV and DT calculation stops, when the 1st product (any of the products) has the fix verified by customer.

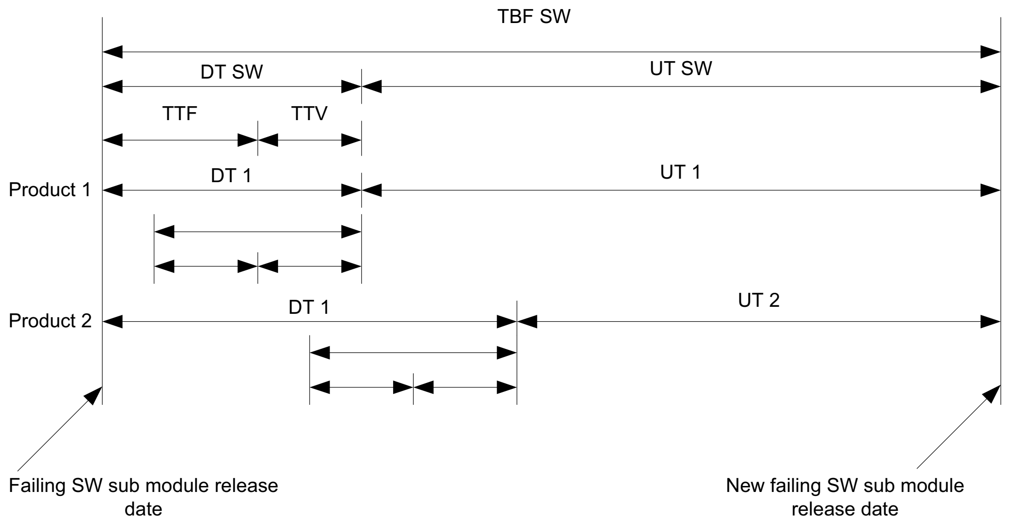

TBF SW
DT SW
UT SW
TTF
TTV
Product 1
DT 1
UT 1
Product 2
DT 1
UT 2
Failing SW sub module release date
New failing SW sub module release date

Figure 16: Down time split in SW module level analysis.

When calculating Mean Up Time (MUT) and Mean Down Time (MDT) for SW module or test area, the evaluation should be done for all incidence reports to that SW sub module or test area. Thus, if the situation is as described in Figure 14 DT 1, UT 1, DT 2, DT 2 should be used instead of DT SW and UT SW.

DT SW and UT SW are valid for single SW sub module generation only. These can be used along with alarm limits to improve fixing speed in active program development time or for later analysis purposes to pinpoint possible problem areas in that context.

MTBF for a SW module should be calculated from all TBF SW values over time. Thus the split between products is not implemented for this calculation.

5.2.1 Example for MTBF calculation for a SW module

In end of year (31.12.2008) customer interface group has to make a yearly report including MTBF calculations. Let’s assume that for a SW module we have detected a single root cause for all incidence reports reported to it and there is three incidence reports reported for it.

Incidence reports 1 and 2 are reported to a SW module version (v1.0) created 22.04.2008 and report 3 is reported for SW module version (v2.0) created 17.06.2008. These dates are also down time start dates for mentioned versions of the SW module.

Report 1 and 2 are created 09.05.2008 and Report 3 19.06.2008.
Report 1, report 2 and report 3 have been fixed respectively in 14.05.2008, 19.05.2008 and 21.09.2008. Verification dates for the reports are respectively 28.05.2008, 23.05.2008 and 21.09.2008.

31.12.08

	SW	DT SW	DT #	Fixed	Verified
R1	22.4.08	22.4.08	9.5.08	14.5.08	28.5.08
R2	22.4.08	22.4.08	9.5.08	19.5.08	23.5.08
R3	17.6.08	17.6.08	19.6.08	21.9.08	21.9.08

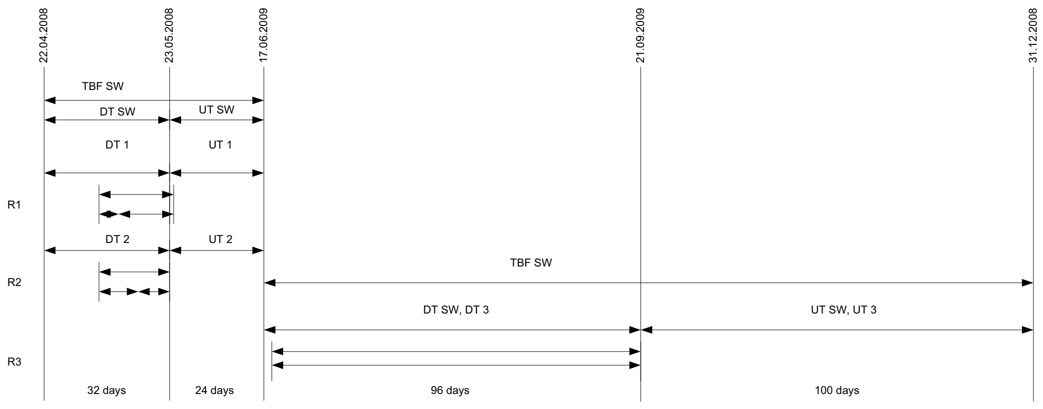


Figure 17: MTBF for a SW module

From Figure 17 it can be seen, that *first verification* for SW module version 1.0 is 23.05.2008 – or 32 days after creation date of that. As next SW module version to have an incidence report is version 2.0 that is created in 17.06.2008:

- DT for version 1.0 is 32 days
- UT for version 1.0 is 24 days
- TBF for version 1.0 is 56 days

Report 3 has been reported for SW module version 2.0 and no further reports have done for this SW module. Therefore:

- Reporting date (31.12.2008) will be used as TBF end date for version 2.0, thus TBF for version 2.0 is 196 days
- DT for version 2.0 is 96 days
- UT for version 2.0 is 100 days

Therefore MTBF for this SW module is:

$$MTBF = \sum_{i=1}^{n} \frac{TBF_i}{n} = \frac{(56+196)days}{2} = 126 days$$

From previous it can be seen, that MTBF is constantly living value that is depended on the reporting date. It can also be calculated that in customer perspective up time ratio $UT(\%)$ for this SW module in end of year 2008 would be:

$$UT(\%) = \sum_{i=1}^{n} \frac{UT_i}{TBF_i} = \frac{(24+100)days}{(56+196)days} = 49{,}2\%$$

5.2.2 Comparable MTBF related values

Different DT results may have different maximum allowed processing time, but the results should be comparable between each others. In this case, if comparable results should be created, each product or SW module should have maximum Allowed Handling Time (AHT) defined for it. The comparable DT (cDT) value will be defined by following formula:

$$cDT = \frac{DT}{AHT}$$

The same principle can be used for UTs and TBFs, also. Thus following formulas could be used for calculating comparable values for those:

$$cUT = \frac{DU}{AHT}$$

$$cTBF = \frac{TBF}{AHT}$$

These comparable values can be used to create uniform metric to all products, SW layers or applications.

6 Findings

The basis of this thesis was to comprehend SW as functional module, that can live during time. Thus SW modules can also wear out, albeit abruptly (versus mechanical wearing). Therefore MTBF is possible to be calculated. Indication of such failure is a customer incidence report. Every new version platform SW, application or sub module is possible cause of new error. It is task of FMEA analysis to define, if the new report is created for new or already known root cause.

In this thesis RCM process is implemented in such way, that SW MTBF is possible to measure in sub module level and also over platform SW releases at the same time and in the way that the statistic run for different applications, SW modules or products is comparable. That is totally new approach in this field. Because in SW development environment the test cases are tightly bind to functions, MTBF is possible to be calculated in test case super group level, but usually not in test case level.

When a customer incidence report is received it is already known, that customer interface team has to react at least with communication (failure effect) to the customer direction and the consequence of the incidence report is at least economical. However, during FMEA analysis both consequence and effect may be upgraded.

Down time split to Time To Fix and Time To Verify enables to gather statistics where the bottlenecks of fix roll out are; is the bottle neck in implementation or verification phase. If this statistics is gathered, also sorting according those is automatically possible.

Current data base system does not need big changes, if RCM approach will be taken into usage in customer interface. Most of the changes can be managed by adding supporting fields not currently present in the data base and adding those to the list of fields to be exported to the external data analysis tool. The bigger modification requirements are imposed to the analysis tools, than data base structure itself.

Hidden or safety category incidence reports were not found in the analysis done to real data after this thesis. Hidden incidence reports can exist only for internal testing results, as customer will be always aware of the results of its own testing. In that sense Hidden Safety and Hidden Economical MSI categories are not valid for direct customer feedback. Detectable Safety category incidence reports do not exist, if the incidence reports to be analyzed are focused to an application that is not related to safety related functions of the device – and no interference with safety functions is to be found during the FMEA analysis.

Combined history of RCM and MSG is not gathered together as completely in any document as in this thesis.

7 Benefits of implementation

FMEA analysis increases customer understanding. I found out, that my customer understanding improved even I have been working on this position already for four years. That is achieved, because after FMEA analysis it can determined, which test cases are failing for which root cause – or is a test case failing because of multitude of root causes.

Quality of maintenance tasks will be greatly increased, when FMEA is run to the customer reports. From FMEA analysis it is possible to define MSIs and maintenance task list for root causes. After FMEA the task list can be ordered according root cause impact in stead of failing test case impact. Proper root cause analysis can also reveal initial wrong verdict for customer incidence reports.

Benefit from the RCM style maintenance task list is, that customer satisfaction will be increased. Communication package (maintenance task list) is good tool to pinpoint most critical improvement areas to R&D and to the customer. After delivering it every one involved in the process knows, why some test case is not passing and what would be benefits or effort to fix it. The root causes can be sorted not only by *failure consequences* or *failure effects*, but also according the amount of incidence reports the root causes are contributing giving better organized priority listing than using only one of those attributes for prioritizing. Customer or customer interface can more effectively prioritize a redesign requests for a root cause instead of failing test case, when maintenance task list is properly organized.

Implementing RCM style fields and functions to the incidence report handling database would enable collecting of new statistical data that would show customer view on SW in numerical format. Also the silent information collected by different customer interface group members would become easily handled public domain as anyone could refer to data base reports containing *maintenance tasks* when making analysis of new *incidence report*. By using data base, the data would also be coherent in all extracted reports compared to spreadsheet / word processor approach used in the first analysis that was done using this thesis as guide line.

8 Further development

This thesis is a good base to implement real functional relation data base for RCM compliant data base for SW development purposes. Improving the statistical tools and reporting functions to an existing or new data base system would be also good development item. Adding the web frontend as data base UI for the data base backed with server side scripting language, such as PERL, Ajax or PHP should be considered.

In Nokia extending RCM analysis to other applications is one important development activity. Also training material for RCM should be created, if RCM will be adopted in Nokia Devices. Naturally this kind of activities can be good themes for a B.Sc. thesis work even thesis is not commissioned by Nokia.

9 List of references

Moubray, John 1997, Reliability-centered Maintenance (2nd Edition), ISBN 0-7506-3358-1

Friend, C. H. 1992, Aircraft Maintenance Management, ISBN 0-582-03866-9

S9081-AB-GIB-010 Rev1 18.04.2007, Military Standard, Reliability-Centered Maintenance (RCM) Handbook [pdf-file] [referred 05.02.2009] *http://www.everyspec.com/USN/NAVSEA/download.php?spec=NAVSEA_RCM_Handbook_DTD_18_April_2007.006051.pdf*

NAVAIR 00-24-403 01.06.2005, Management Manual, Guidelines for the Naval Aviation Reliability-Centered Maintenance Process [pdf-file] [referred 05.02.2009] *http://www.barringer1.com/mil_files/NAVAIR-00-25-403.pdf*

Järviö, Jorma 17.11.2004, Ehkäisevän kunnossapidon suunnittelu [pdf-file] [referred 05.02.2009] *http://ylivieska.cop.fi/sjjkurssit/kupitek/sis%C3%A4lt%C3%B62008/Kunnossapitostrategia/RCM%20Jarvio%202004.pdf*

NASA-RCM-267 28.02.2000, NASA Reliability Centered Maintenance Guide for Facilities and Collateral Equipment [pdf-file] [referred 05.02.2009] *http://www.everyspec.com/NASA/NASA+(General)/download.php?spec=NASA_RCM.267.pdf*

NAVAIR 14.12.1998, Reliability centered maintenance [ppt-file] [referred 05.02.2009] *http://www.navair.navy.mil/logistics/rcm/library/Mgmt%20Brief.ppt*

MIL-STD-2173(AS) 21.01.1986, Military Standard, Reliability-Centered maintenance [pdf-file] [referred 05.02.2009] *http://www.weibull.com/mil_std/mil_std_2173.pdf*

MIL-STD-1843 (USAF), 08.02.1985, Military Standard, Reliability-Centered Maintenance for Aircraft, Engines and Equipment [pdf-file] [referred 05.02.2009] *http://www.barringer1.com/mil_files/MIL-STD-1843.pdf*

AD-A066579, 29.12.1978, Reliability-Centered Maintenance [pdf-file] [referred 05.02.2009] *http://www.barringer1.com/mil_files/AD-A066579.pdf*

APPENDIX I: INCIDENCE REPORT FIELDS EXAMPLES
(excluding incidence report body and title)

ISSUE			TEST CASE		SW INFO					TO MTBF					FMEA		
Product	ID	Created	Test Case	Area	SW Detected	Devicee SW date	Platform 1 date	Platform 2 date	…	Fixed SW	Fix verification date	Duration (days)	TTF (days)	TTV (days)	Failure	Failure Mode	Failure Mode LV2 (1,2,3,…)
Product 1	1	17.6.08	1.1.18	Dial up	1.2	22.5.08	7.3.08	15.2.08		n/a					4 Test Case is not passed	Server	A1 Configuration
Product 1	2	17.6.08	1.1.19	Dial up	1.2	22.5.08	7.3.08	15.2.08		n/a					1 Test Case can not be run	Server	A2 Malfunction
Product 1	3	17.6.08	1.2.1.10	Data transfer	1.2	22.5.08	7.3.08	15.2.08		n/a					4 Test Case is not passed	Server	A3 Funtionality Interference
Product 1	4	17.6.08	5.7.12	Data Conversion	1.2	22.5.08	7.3.08	15.2.08		n/a					4 Test Case is not passed	Test Case	B1 Test Case Direct Flaw
Product 1	5	17.6.08	5.7.15	Data Conversion	1.2	22.5.08	7.3.08	15.2.08		n/a					5 Test Case passed, but should fail	Test Case	B2 Test Case Content Not Supported
Product 1	6	18.6.08	2.3.4.15	FTP	1.2	22.5.08	7.3.08	15.2.08		n/a					4 Test Case is not passed	Test Case	B3 Test Case Instruction Not OK
Product 1	7	18.6.08	5.7.22	Data Conversion	1.2	22.5.08	7.3.08	15.2.08		n/a					4 Test Case is not passed	Test Case	B4 Test Case Indirect Flaw
Product 1	8	18.6.08	1.2.30	Dial up	1.2	22.5.08	7.3.08	15.2.08		n/a					4 Test Case is not passed	Human	C1 Test Setup
Product 2	9	17.7.08	10.1.2.30	Video call	1.0	23.6.08	25.4.08	15.2.08		n/a					3 Test Case is passed paritally	Human	C2 Test Case Instruction Not Understood
Product 2	10	17.7.08	3.3.15	Voice call	1.0	23.6.08	25.4.08	15.2.08		n/a					4 Test Case is not passed	Human	C3 Device Capabilities Not Understood
Product 2	11	17.7.08	3.3.16	Voice call	1.0	23.6.08	25.4.08	15.2.08		n/a					1 Test Case can not be run	Human	C4 Device Capabilities Not Available (early SW)
Product 2	12	24.7.08	3.3.17	Voice call	1.0	23.6.08	25.4.08	15.2.08		n/a					4 Test Case is not passed	Human	C5 Wrong Specification Used
Product 2	13	24.7.08	3.3.18	Voice call	1.0	23.6.08	25.4.08	15.2.08		n/a					4 Test Case is not passed	Human	C6 Wrong Test Case Verdict
Product 2	14	24.7.08	3.3.19	Voice call	1.0	23.6.08	25.4.08	15.2.08		Product	3.8.08	41	35	6	4 Test Case is not passed	Integration	D1 Component Missing - Variant
Product 2	15	24.7.08	2.3.4.5	FTP	1.0	23.6.08	25.4.08	15.2.08		Product	1.8.08	39	25	14	4 Test Case is not passed	Integration	D2 Component Fail - Variant
Product 2	16	24.7.08	2.3.4.6	FTP	1.0	23.6.08	25.4.08	15.2.08		Product	1.8.08	39	20	19	4 Test Case is not passed	Integration	D3 Component Configuration - Variant
Product 2	17	25.8.08	1.1.18	Dial up	1.0	23.6.08	25.4.08	15.2.08		Product	1.8.08	39	15	24	4 Test Case is not passed	Integration	D4 Component Upgrade not done - Variant
Product 2	18	25.8.08	1.2.30	Dial up	1.0	23.6.08	25.4.08	15.2.08		Product	1.8.08	39	17	22	4 Test Case is not passed	Integration	D5 Component Missing - Core
Product 2	19	17.8.08	1.15.23	Dial up	1.0	23.6.08	25.4.08	15.2.08		Platform1	19.9.08	147	111	36	4 Test Case is not passed	Integration	D6 Component Fail - Core
Product 3	20	17.8.08	1.2.28	Dial up	1.0	2.8.08	25.4.08	27.5.08		Product	19.9.08	48	18	30	4 Test Case is not passed	Integration	D7 Component Upgrade not done - Core
Product 3	21	17.8.08	1.2.29	Dial up	1.0	2.8.08	25.4.08	27.5.08		Product	19.9.08	48	18	30	4 Test Case is not passed	Integration	D8 Component Configuration - Core
Product 3	22	17.8.08	1.2.30	Dial up	1.0	2.8.08	25.4.08	27.5.08		n/a					4 Test Case is not passed	Support	E1 Deprechiated feature
Product 3	23	17.8.08	1.2.31	Dial up	1.0	2.8.08	25.4.08	27.5.08		n/a					4 Test Case is not passed	Support	E2 Higher or adjasent module feature
Product 3	24	19.8.08	1.10.5	Dial up	1.0	2.8.08	25.4.08	27.5.08		n/a					4 Test Case is not passed	Support	E3 New feature request (not supported)
Product 3	25	20.8.08	1.15.23	Dial up	1.0	2.8.08	25.4.08	27.5.08		Platform1	19.9.08	147	111	36	5 Test Case passed, but should fail	Support	E4 Partial support only

		FAILURE CONSEQUENCE				TASK		
Failure Mode LV3 (A,B,C,...)	Unique ID	Hidden	Safety	Operational	Economical	Type	Highest task class (Failure effect)	Description
A Server parser has to be reconfigured	4A1A	NO	NO	YES		DO	2 Configuration	Reconfigure parser
A Server was down	1A2A	NO	NO	YES		DO	1 Communications	Inform customer, that server was down, when this test case was run
A Server was busy	4A3A	NO	NO	YES		DO	4 Integration	Increase server capability with new CPU and interface units
A Error in test case code	4B1A	NO	NO	YES		DO	3 Test Case redesign	Fix error in test case 5.7.12 in line 10. Inform customer
A Content file WMA 192 kHz	5B2A	NO	NO	NO	YES	DE	1 Communications	Infrom customer, that WMA 192 kHz file is not supported by device
A Conflicting instruction vs. test case code	4B3A	NO	NO	NO	YES	DE	1 Communications	Infrom customer, that this test case haS flaw in the instruction
A Call method	4B4A	NO	NO	NO	YES	DE	1 Communications	Infrom customer that the feature that is supposed to be tested is supported, but the unsupported call method causes the fail.
A Hands free on	4C1A	NO	NO	NO	YES	DE	1 Communications	Infrom customer that the device should be in hand set (HS) mode, not in hands free (HF) mode in this test case
A Test case 10.1.2.30 step 5	3C2A	NO	NO	NO	YES	DE	1 Communications	Inform customer that test case step 5 was not run according the instructions
A Design Limitation	4C3A	NO	NO	NO	YES	DE	1 Communications	Inform customer that this 3rd party properiarity feature is not supported in our devices
A Release notes	1C4A	NO	NO	NO	YES	DE	1 Communications	Inform customer that this limitation was mentioned in release notes, will be supporeted in sales SW
A VOX v1.7	4C5A	NO	NO	YES		DO	1 Communications	Inform customer, that our devices support VOX v1.3 currently
A Vrong Verdict	4C6A	NO	NO	NO	YES	DE	1 Communications	Inform customer, that according the provided information and test case this verdisct should be PASSED
A Implementation was not done	4D1A	NO	NO	YES		DO	4 Integration	Inform customer, that this feature will be properly integrated in next delivery. Verify in next release before delivery.
A Application fails	4D2A	NO	NO	NO	YES	DE	4 Integration	Inform customer, that application delivered by them is failing. New application to be integrated, when available.
A Customer configuration not done	4D3A	NO	NO	YES		DO	2 Configuration	Inform customer, that this feature will be properly configured in next delivery. Verify in next release before delivery.
A Variant version	4D4A	NO	NO	YES		DO	4 Integration	Inform customer, that revised version of variant will be integrated in next delivery. Verify in next release before delivery.
A Implementation was not done	4D5A	NO	NO	YES		DO	4 Integration	Inform customer, that this feature will be integrated in next delivery. Verify in next release before delivery.
A Telephony application fails	4D6A	NO	NO	NO	YES	DE	5 Redesign	Inform customer, that in this case Telephony application fails. Fix will be integrated in sales SW.
A Customer configuration not done	4D7A	NO	NO	YES		DO	2 Configuration	Inform customer, that this feature will be properly configured in next delivery. Verify in next release before delivery.
A Variant version	4D8A	NO	NO	YES		DO	4 Integration	Inform customer, that revised version of variant will be integrated in next delivery. Verify in next release before delivery.
A Deprechiated	4E1A	NO	NO	NO	YES	DE	1 Communications	Inform customer, that this feature is not supported in this category devices
A Supported by Camera application	4E2A	NO	NO	NO	YES	DE	1 Communications	Inform customer, that in our devices this is supported in other application and this test case should be tested with it
A Redesign reguest	4E3A	NO	NO	YES		DO	5 Redesign	Inform customer, that their redesign request will be forwarded to R&D. Create redesign request.
A Connection re-establishment after voice ca	5E4A	NO	NO	YES		DO	5 Redesign	Inform customer, that we do not have support for this particular method. Redesign ongoing.

APPENDIX II: Customer communications packet template

CTA Customer Communications Packet for Customer: Application

Change history:

Version	Date	Status	Comments / Author
0.1	09.12.2008	Draft	First draft version / Mika Peltokorpi
0.2	15.01.2009	Proposal	First proposal version / Mika Peltokorpi
0.3	03.02.2009	Proposal	2nd Proposal / Mika Peltokorpi

Table of contents:

1. Introduction 4

2. Actions 5

2.1 Functional changes in version 2.0 5

2.1.1 Version 2.0 roll-out plan 5

2.2 Communication tasks 6

2.2.1 WMA 192 kHz 6

2.3 Configuration tasks 6

2.3.1 Parser configuration 6

2.4 Test case redesign tasks 6

2.4.1 Test Case 5.7.12 code error 6

2.5 Integration tasks 6

2.5.1 Implementation of *this feature* was not done 6

2.6 Redesign tasks 6

2.6.1 Connection re-establishment after voice call 6

1. Introduction

This report is based on FMEA analysis results for Application issue reports. FMEA analysis has been made for all issues reported to all Nokia products reported until 31st Jun 2008.

2. Actions

2.1 Functional changes in version 2.0

Following functional changes are implemented to version 2.0 of Application, which is rolled out to Nokia devices starting from Q2/09:

- Change 1
- Change 2
- …

2.1.1 Version 2.0 roll-out plan

Product	**The first Product SW release to have Application v2.0**
Product 1	3.0
Product 2	2.1
Product 3	1.4
…	

2.2 Communication tasks

2.2.1 WMA 192 kHz

Failure reason: WMA 192 kHz is not supported

Area: Data conversion

Failure ID: 5B4A

Consequence: Detectable Economical

Occurrence: 1

Task category: Communication

- Our devices do not support WMA 192 kHz sample rate

Issues reported: 5

Test Cases affected: 5.7.15

2.3 Configuration tasks

2.3.1 Parser configuration

Failure reason: Server parser configuration error

Area: Dial up

Failure ID: 4A1A

Consequence: Detectable Operational

Occurrence: 1

Task category: Configuration

- Server parser function had configuration error that was corrected by replacing lines 15 to 20 with following code:

Task category: Communication

- Inform customer, that this was server related error that has been fixed.

Issues reported: 1

Test Cases affected: 1.1.18

2.4 Test case redesign tasks

2.4.1 Test Case 5.7.12 code error

Failure reason: Test case 5.7.12 line 10
Area: Data conversion
Failure ID: 4B1A
Consequence: Detectable Operational
Occurrence: 1

Task category: Test case redesign

- Test case 5.7.12 line 10 has to be replaced with following code:

Task category: Communication

- Inform customer, that this test case had error in line 10 and it has been fixed now.

Issues reported: 4

Test Cases affected: 5.7.12

2.5 Integration tasks

2.5.1 Implementation of *this feature* was not done

Failure reason: Integration of *this feature* is coming to product SW 1.2

Area: Dial up

Failure ID: 4D5A

Consequence: Detectable Operational

Occurrence: 1

Task category: Integration

- Verify, that the inmtegration of *this feature* is working on SW 1.2

Task category: Communication

- Infrom customer, that *this feature* will be implemented in SW 1.2

Issues reported: 18

Test Cases affected: 1.2.30

2.6 Redesign tasks

2.6.1 Connection re-establishment after voice call

Failure reason: Connection re-establishment after voice call, metod not supported

Area: Dial up

Failure ID: 5E4A

Consequence: Detectable Operational

Occurrence: 1

Task category: Redesign

- Change request had been already created. Redesign ID: 11523-08. Followup the redesign process.

Task category: Communication

- Inform customer, that this optional call method in the specification is not currently supported by our device, but it will be supported during this year. Detailled roll out plan will be informed later.

Issues reported: 25

Test Cases affected: 1.15.23