TAMPERE
POLYTECHNIC

FINAL THESIS

Customer Technical Acceptance issue handling optimization

Marko Humalisto

Business Information Systems
05 2007
Supervisor: Maritta Hoffrén

TAMPERE 2007

TAMPERE
POLYTECHNIC

Author(s) Marko Humalisto

Degree Programme(s) Business Information Systems

Title Customer Technical Acceptance issue handling optimisation

Month and year 5/2007

Supervisor Maritta Hoffrén **Pages:** 50

ABSTRACT

This thesis has been requested by Tampere Nokia Multimedia. Its purpose was to investigate current Multimedia Customer Technical Acceptance (CTA) error handling process and tools during acceptance testing. Error as a concept is called at this stage as an issue until it will be validated to be a valid error.

Mobile phones have to go through many phases before those will end up to retailer and end users. Nokia customers in this case, are seen as operators or other smaller distribution channels. One of the latest phases before sales is the product approval, which means approval for a product by customer before Nokia can start the sales.

Customer acceptance testing is relevant for product approval because it can ensure adequate quality of a product for customers before they are making the final sales decision. Customers are testing a product against their own requirement documents, possible anomalies and issues will be reported to manufacturer.

Customers are reporting issues according to the Nokia issue handling process and with the help of agreed tools. Reported issues will be handled through the Nokia issue handling team and validated errors will be transferred forward to Nokia Research & Development (R&D), where those are investigated and corrected. At this level, CTA ensures that, issues which are preventing the sales are corrected, so that customer could give the approval for a product and Nokia could start the sales to the customers.

The purpose of this thesis is to wrap up current issue handling process of Nokia Multimedia CTA and tools used for that. The target is to find problematic areas and make proposals to correct them.

During the thesis construction, advantage of own work experience about CTA customer approvals and issue handling were taken. Knowledge of colleagues was used as well. In addition to that, methods of process development, fundamental knowledge of software engineering and testing were in an important role.

As a result, current issue handling process was documented and problematic issues of the process were mapped. According these results, it was possible to make improvement to overall process as well as several smaller changes. The target of the proposal is to eliminate most serious problems, to adapt the process, make tools to meet current requirements and making the issue handling more effective during the acceptance testing.

Keywords Error management Issue management Acceptance testing Customer Nokia

TAMPEREEN
AMMATTIKORKEAKOULU

Tekijä(t) Marko Humalisto

Koulutusohjelma(t) Tietojenkäsittely

Tutkintotyön nimi Customer Technical Acceptance issue handling optimisation

Työn valmistumis-kuukausi ja -vuosi 5/2007

Työn ohjaaja Maritta Hoffrén **Sivumäärä:** 50

TIIVISTELMÄ

Tampereen Nokian Multimedia -yksikkö tilasi tämän opinnäytetyön, jonka tarkoitus on tarkastella nykyistä Multimedia -yksikön asiakashyväksyntätiimin (Customer Technical Acceptance, CTA) virheenkäsittelyprosessia ja siinä käytettäviä työkaluja asiakashyväksyntätestauksen aikana. Virhettä kutsutaan vielä tässä vaiheessa ongelmaksi (issue), kunnes se on todennettu oikeaksi virheeksi (error).

Matkapuhelinten tekemisessä on monta eri vaihetta ennen kuin tuotteet päätyvät jälleenmyyjille ja loppuasiakkaille. Nokian asiakkaina ovat usein operaattorit tai muut pienet jakelukanavat. Asiakashyväksyntä on tuotekehityksen viimeinen vaihe ennen kuin tuotteen myynti aloitetaan.

Asiakashyväksyntään liittyy olennaisesti tuotteen hyväksymistestaus, jolla asiakkaat voivat varmistaa tuotteen riittävän laadun ennen lopullista ostopäätöstä. Asiakkaat testaavat tuotetta vertaamalla sitä vaatimuksiinsa, ja mahdolliset poikkeamat ja virheet raportoidaan tuotteen valmistajalle.

Asiakkaat raportoivat poikkeamat Nokian virheenkäsittelyprosessin mukaisesti sovittujen työkalujen avulla. Raportoidut virheet käsitellään virheenkäsittelytiimin kautta ja validiksi todetut virheet syötetään edelleen Nokian tuotekehityksen tutkittavaksi ja korjattavaksi. Tässä vaiheessa asiakashyväksyntätiimi varmistaa, että asiakkaiden myynnin esteenä olevat poikkeamat korjataan, jotta asiakas antaisi tuotteen asiakashyväksynnän ja Nokia voisi aloittaa myynnin asiakkaalle.

Tämän opinnäytetyön tarkoituksena oli kartoittaa nykyinen Nokia Multimedia -asiakashyväksyntätiimin virheenkäsittelyprosessi ja siinä käytettävät työkalut. Tavoitteena oli löytää ongelmalliset osa-alueet ja laatia niille parannusehdotukset.

Opinnäytetyön laadinnassa käytettiin hyväksi omaa sekä kollegojen työkokemusta Nokian asiakashyväksynnästä sekä virheenkäsittelystä. Lisäksi hyödynnettiin prosessinkehittämismenetelmiä sekä ohjelmistotuotannon ja testauksen perusoppeja.

Opinnäytetyön lopputuloksena nykyinen virheenkäsittelyprosessi dokumentoitiin sekä prosessin ongelmalliset osa-alueet selvitettiin. Näiden tulosten pohjalta pystyttiin tekemään parannusehdotus, jonka tavoitteena oli poistaa vakavimmat ongelmat, mukauttaa prosessi ja työkalut vastaamaan nykyisiä toimintatapoja sekä tehdä virheenkäsittelystä entistä tehokkaampi asiakashyväksynnän välivaihe.

Avainsanat Error management Issue management Acceptance testing Customer Nokia

Table of contents

1 Abbreviations

CTA = Customer Technical Acceptance
Error = Incorrect action or calculation performed by software
ES = Enterprise Solution
HSDPA = High Speed Downlink Packet Access
Issue = An unidentified problem
NMP = Nokia Mobile Phones
MP = Mobile Phones
OTC = Operator Technical Co-operation
PCP = Product Creation Process
PM = Product Marketing
R&D = Research and development
SW = Software
TP = Technology Platforms
TRUE = Trusted & Rewarding User Experience
TSW = Terminal Software

2 Introduction

2.1 Starting point

Actions for Nokia CTA issue handling optimization started after the current process was found ineffective. Customers were not satisfied with Nokia's issue feedback and service in general, during the product acceptance phase. There was also a negative feedback from stakeholders, that the customer acceptance work was insufficient. According the customers and Nokia interfaces, issues were not handled fast enough within the required time frame. This had direct negative influence on technical acceptance and quality of service. In addition, it was important to get issue reports from customers to the Nokia system during the customer acceptance testing and it was also necessary to get critical errors pushed to *Nokia Program* into the sw correction process. Once customers reported the issues, they were expecting fast feedback related to the possible error fixing solutions and schedules.

The information supposed to be provided for the customers within certain time, was always delayed, and its content was not accurate. The bottle neck for this was the issue handling, resources, tools, missing documentation and ineffective issue management process. Some of the problems were also because of the lack of proper training for issue management work. It was found that documents and clear process were missing for issue handling. Tools were found to be inefficient internally as well. There were occasional problems with tools and customer service was not properly taken care of. This thesis is focusing on problems in CTA issue handling process and tools which are used for that. Preliminary work for development work started in 2004.

2.2 Overview

First, everything starts with Nokia history and organization structure. To understand Nokia's product acceptance phases, readers have to know the basic idea of *Nokia Multimedia* customer technical acceptance and how it is co-operating with different kind of stakeholders. It is also important to know the background of the most common tools that CTA is using. These are covered in chapters 3 and 4.

In chapter 5, there are explained fundamental issues of the software testing. The most relevant examination from Nokia perspective is the difference between an issue and an error. Acceptance testing in general is only explained lightly, since the topic is covered more thoroughly in the chapter 6, where the whole CTA acceptance testing is gone through.

Chapters 7, 8 and 9 are getting their teeth into the actual problem and development work what this thesis is covering up. The old CTA error handling work is presented in chapter 7, and chapter 8 is identifying the main problems of the old way of working and making new proposals for fixing them. Chapter 9 is presenting the improvements which are the results of identifying the problems. The whole thesis will be wrapped up in the last chapter 10, where it is considered how well the development work was succeed, what problems were fixed and which issues needed more development work.

3 Introducing Nokia

3.1 History

Nokia has a long history starting in 1865, when forest industry enterprise was established in Tampere. Before Nokia found its current business area, telecommunications and mobile phones, there were also diverse other business areas involved through the 20th century. After manufacturing paper, Nokia was involved in rubber industry, which also led to the establishment of the Finnish Cable Works.

Cable industry was paving the way forward to telecommunications and even further to computer manufacturing in the 80's. Since the beginning of the 1990's, Nokia has concentrated on telecommunications, by divesting its information technology and basic industry operations.

The company includes nowadays four business groups: *Mobile Phones (MP)*, *Multimedia*, *Enterprise Solutions (ES)* and *Networks.* Nokia also includes two horizontal groups that support the business groups: *Customer and Market Operations* and *Technology Platforms (TP)* (Nokia website: Nokia Business Group Structure).

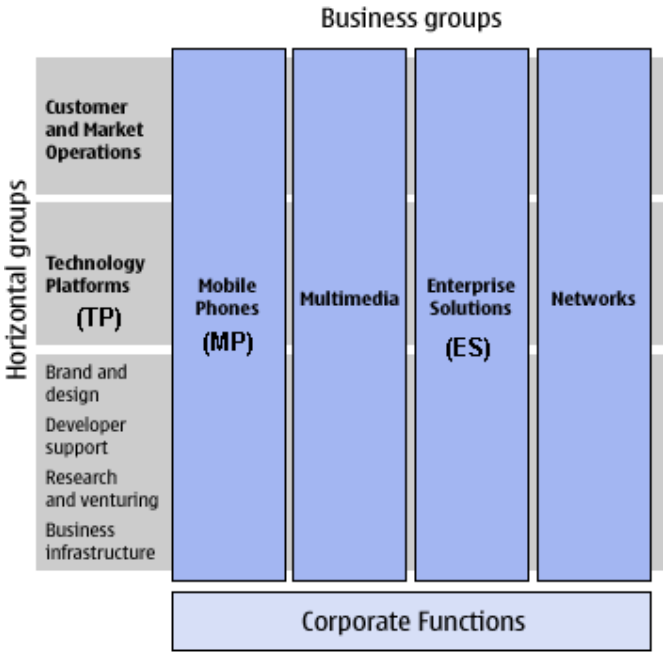

Figure 1 Nokia business groups and horizontal groups (Nokia website: Nokia Business Group Structure)

3.2 Customer and market operations

There are sales, marketing, sourcing, manufacturing and logistics included in *Customer and Marker Operations* supporting 3 business groups; *Mobile Phones*, *Multimedia* and

Enterprise Solutions. *The Networks business group* has its own dedicated sales and marketing, logistics and sourcing functions (Nokia website: Customer and Market Operations).

3.3 Technology Platforms

Technology Platforms (TP) is responsible for Nokia's technology and competitiveness. It is providing leading technologies to platforms to Nokia's business groups as well as external customers. E.g. S60 and S80 are platforms provided by TP. S60, used mostly in multimedia phones, S80 is a platform used in business phones, such as Communicator (Nokia website: Technology Platforms).

3.4 Mobile Phones

Mobile Phones products are focused primary on the markets of large volumes and on the main streams of consumers. It has been put more effort to the design, facility of use and the low price. In addition, some basic functions, have been added like basic cameras, music players and high quality screens, which are interests for most of the people nowadays (Nokia website: Mobile Phones).

3.5 Multimedia

Multimedia products are focused especially to the multimedia functions. High quality cameras, video calls, high quality media players and good internet browser are features which will clearly jump out of the technology mainstream. Multimedia is using S60 platform in their products (Nokia website: Multimedia).

3.6 Enterprise Solutions

Interest of *Enterprise Solution* is to provide products and security infrastructure solutions to business world. Products are focused to optimize the usage fixed IP network security, mobilize corporate email and extend corporate telephone systems to Nokia's mobile devices (Nokia website: Enterprise Solutions).

3.7 Networks

Networks, provides network infrastructure, communications and networks service platforms, as well as professional services to the operators and service providers. Currently in network side, GSM (2G), EDGE (2.5G) are used, but little by little the whole world is heading towards wideband technologies, which provides more information capacity in the networks for the users. So called 3G/WCDMA and HSDPA technologies are now breaking through in the industrial countries and more likely developing countries are following the trend (Nokia website: Networks).

4 Multimedia Customer Technical Acceptance

4.1 General

Customer Technical Acceptance (CTA) is part of Nokia Multimedia business group. It is combined from several teams, which are located all around the world. Currently, CTA is led from Tampere, where the Tampere team is also located. Other teams are located all around the world, and those are called as “virtual teams”, once the composition depends on active projects.

The purpose of CTA is to ensure efficient technical customer acceptance for Multimedia products in time for production and sales readiness. CTA is also doing close co-operation with the customers such as: testing and verification for a software quality. Multimedia CTA is only responsible for Multimedia product acceptance, since CTA is a part of the Multimedia business group. Other Business Groups have a similar responsibility.

CTA is interoperating between customers and Nokia programs. CTA is providing information about software quality supplied by the customers for Nokia Research and Development (R&D). Customers are testing hardware and software quality (functionalities) and reporting the issues by inserting issue reports in *OTC TRUE* database system. Issue reports are handled and pre-filtered by CTA issue management team, through the TRUE database system which service *OTC (Operator Technical Co-operation)* organization is providing. All the valid errors are transferred forward to R&D, where the needed corrections are done.

4.2 Relevant stakeholders

Customer Acceptance team works together with several stakeholder groups. In addition to that, CTA is interacting with several parties inside its own business group Multimedia. There is also co-operation with other business groups: ES, MP and TP as well, and some stakeholders which are outside from Business Groups.

4.2.1 CTA customers

Main customers are the biggest operators, who usually have the highest sales volumes. The rest of the customers are consistent of other operators, regions, countries and retailers. Consumers and end users are not included in CTA customer scope, but covered by other Nokia functions.

4.2.2 Technology Platforms

Technology Platforms is providing the software platform for Nokia products. Multimedia phones uses *S60 platform* which is using *Symbian* operating system. S60 is developed primarily by Nokia, and it is licensed also to other manufacturers. *Platforms* are providing their platform software for *programs* which are finalizing the product by integrating their own specific product software in it. *Series 40* and *Series 80* are included to the platform family (Figure 2).

CTA interacts primarily with *product programs*, when related to software issues but, sometimes, *S60 platform* will be contacted directly. This can happen when it is obvious that customer issues are caused by an issue in S60 implementation, and CTA can help *program* in the error handling work. Then issues are communicated directly to the *S60 platform*. *S60* has its own R&D where the issues are handled.

CTA uses the software information provided by *S60* for planning own activities together with *product program*. Every *program* is using a certain platform version and this has to be known before sw can be integrated. Usually, development for the new sw platform is already ongoing when the old platform releases versions are in the same time developed or ramped down.

Figure 2 Nokia Platform technologies (Nokia Forum: Platforms and operating systems)

4.2.3 Product Program

Product programs are part of business groups. *Product program* means practically resources which are doing the integration and implementation for one product. *Program* is planning the schedule and setting the strategy. It is in charge that targets are achieved.

CTA plans together with the *program* the product acceptance testing, and the vital deadlines when the product needs to be ready. Practically, planning will be daily work until customer acceptance is achieved. *The program* is providing terminals, testing equipment and technical support to the CTA. Part of the terminals is allocated to the customers, for acceptance testing purposes.

Like *S60*, *program* has the R&D department as well. *The program R&D* is only responsible for the specific issues and errors which are not related to *S60 platform* software. *Program R&D* will do error code fixes in the program specific software which is build on the top of *platform* software. *Program R&D* has to take care that all the necessary *platform* fixes are also implemented to the product software.

Program R&D will be the first contacted when it is about correcting the customer related issues and errors for a specific product. CTA will provide customer interface and first level of issue management where issues are first prioritized and filtered before feeding those to *program R&D*. *Program R&D* is responsible to inform CTA about possible error corrections, so that information can be forwarded to customers as well.

4.2.4 Software Variant Team

Multimedia business group has own *variant software team*, which is creating variant software for the customers. Variant creation is done on the top of the basic software, provided from *S60* and *product program*.

Because variant software will be the last software development phase before software is given to customer testing, there will be eventually a certain percentage of variant software errors, which are reported to CTA. That is the reason why there has to be co-operation with variant team, so that customer variant software issues will be fixed and customer acceptance will be achieved in time.

4.2.5 Marketing

Multimedia has its own *marketing organization* concentrated on taking care of marketing activities. In several cases, marketing people are also acting as an interface between customer and Nokia R&D, when related to the technical issues and phone functionalities.

Marketing is trying to sell the products to the potential customers and they are also negotiating about the new requirements and phone functionalities with the customer. If marketing succeed to make a deal with the customer, marketing organization is providing the information to the product program about the phone functionalities and customer requirements. Sometimes requirement are not realistic to implement and requirements need to be re-evaluated.

CTA has to be aware of the customer requirements. Combination of all multimedia products and all the customers which have different kind of setup in their phones can make it complicated. When customers are reporting the issues, it is not enough to check the Nokia technical specifications, but the issues need to be compared to every customer requirement document. Since variant software team is creating customer specific software, marketing needs to co-operate closely with them and create appropriate documents.

4.2.6 Operator Technical Co-operation

Operator Technical Co-operation (OTC) is an organization that provides services and technical support for customers. It is co-operating with operators and customers which have done agreement with OTC organization. Agreement will obligate OTC to serve and give support related to customer issues. Using OTC error database system for reporting issues to Nokia is part of the agreement. OTC is having personnel in the customer interface, which CTA is also using. CTA has a permission to use OTC error database, maintaining and handling customer’s issues.

4.2.7 Regional and Country Product Marketing

Product marketing (PM) is responsible of marketing related issues in different regions and countries all around the world. PM is a channel which is providing issues from regions and countries to CTA. The difference between issue reporting compared to the OTC, is the process how the issues are reported. PM has Nokia internal persons who will report issues based on the internal testing or observations that the customers have seen. So, in that sense, PM testing and reporting is not that organized. However, product acceptance criteria is the same as for other customers and software error corrections are done according the same process than in any other case.

4.3 Tools used in CTA

4.3.1 Lotus Notes

Lotus Notes is a client-server collaborative software and e-mail system owned by *Lotus Software*, of the *IBM Software Group*. *Notes* can be used as variety of personal of business situations. It provides help to users managing daily work e.g. e-mail, schedules, web access or maintaining and storing information. All these functionalities can be built using *Domino environment* and *Domino servers* (Haberman 1999: 4, 6.) From Nokia error management point of view, *Lotus Notes* is used for managing error information in various databases. *Domino servers* and *clients* also enables to flexible work across different time zones all around the world.

4.3.1.1 Domino and Client Servers

In *Notes* database, system information is moving between *Domino servers* and *client servers*. Databases can be located on *Domino* or *client servers*. Databases can be also located on both servers with the same ID, it means that database located in the *client server* will be copy of the database located on *Domino server*. This kind of copy server is caller *replica* (Haberman 1999: 270). Around the *Domino server*, a very complicated network can be build, consisted of many servers and databases. For example: a company could have this kind of server clusters in every country, including various databases. Servers would be connected directly or with the help of other servers to the main *Domino server*. Information would be copied from one server as a chain reaction and databases would always have the latest updated data after the synchronization. Actually, when people are working across different time zones, the data updates between servers will happen all the time.

4.3.1.2 Replication

When data is copied from one server to another, it is called *replication*. During replication, data is synchronized between databases. Synchronization happens in certain periods, which also enable working in offline. Replication encompasses all aspects of a *Notes* database, including data and design. This means that *Notes* databases have a built-in mechanism for deploying application changes to the end users, even remote users who work away from office (Haberman 1999: 59).

4.3.1.3 Database Conflicts

In *Notes* databases, a conflict can happen when multiple users are editing and saving the same document. A conflict can happen in the same database or on multiple replica databases. The conflict can be a *replication* or a *save conflict* (Haberman 1999: 59).

During conflict, *Domino* is using *Notes* document hierarchy solving the problem. *Domino* will store one document as the *main document* and any other documents as *response documents*. Each r*esponse document* will be named either *replication conflict* or *save document*. *Domino* keeps track of the date and time a document is edited to determine which document becomes the *main document* and which document become *response documents* (Haberman 1999: 59).

A replication conflict happens when two or more users edit the same document on different replicas. *Domino* is using a few simple rules to determine the prioritization for saving the *main* and *response document* (Haberman 1999: 59).

The document that has been edited the most will become the *main document* and all others will become *responses*.
If all copies of the documents have been edited in the same number of times, the most recently saved document will become the *main document* and all others will become *responses*.
If one document has been deleted and any other documents have been edited, the deletion will take precedence and all of the documents will be removed (unless the edit took place after the time the document was deleted).

A save conflict happens when two or more users edit the same document on the same server. It does not matter which fields are going to changed, because the first document to be saved becomes the *main document*. After that, any of the other users saving the document will be prompted by a dialog box warning them that the document will be saved as *a save conflict.* If the user chooses the save the document, it will automatically become *a response* to the *main document* (Haberman 1999: 59).

4.3.2 TRUE Database

TRUE database is part of the Nokia Lotes Notes. TRUE system server and network architecture is following Nokia common rules for servers and databases. TRUE database was designed originally for Nokia internal end-user testing, but currently the system is in CTA use as well. The system is maintained by the OTC organization, which is responsible for the development area and solving user's problems related to the tools. The figure 3 is showing a sample view from OTC TRUE database.

The TRUE database concept is used as an error reporting tool for OTC, this tool is referred as OTC TRUE DB and it is a simple and effective way to report errors directly to the *product program.* This will provide a common database for all *product programs.*

TRUE system uses, as other Lotus Notes databases, replication to synchronize data between the servers all around the world. How fast data is copied between the servers depends on the distance of the servers. For example: it takes longer time to replicate from Asia to Finland than from Europe to Finland. To minimize the network traffic, synchronization is done in certain time cycles. In OTC TRUE system, all data is going through the main *Domino server(s)* called *OTC Central*. Data is moving between *Central* and *project (client) servers* which are located as near as possible where the active project is located. *OTC Central* is always between customers and the *projects servers*.

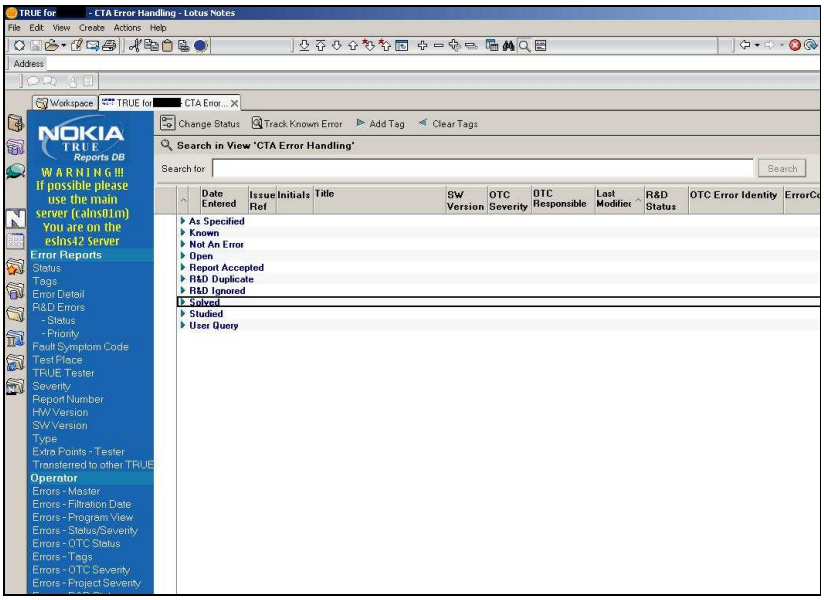

Figure 3 OTC TRUE view

4.3.3 TSW

Originally designed for S60, *TSW (Terminal Software)*, it is a database and an interface to maintain information of the R&D errors. As well as OTC TRUE, TSW is a part of *Nokia Lotus Notes* system. Error analysis is happening in error reports, where all comments and other information is stored. Mainly, it is used for *program* and *S60* software error management. The figure 4 is showing a sample view from TSW database. The actual content related to Nokia internal information cannot be shown.

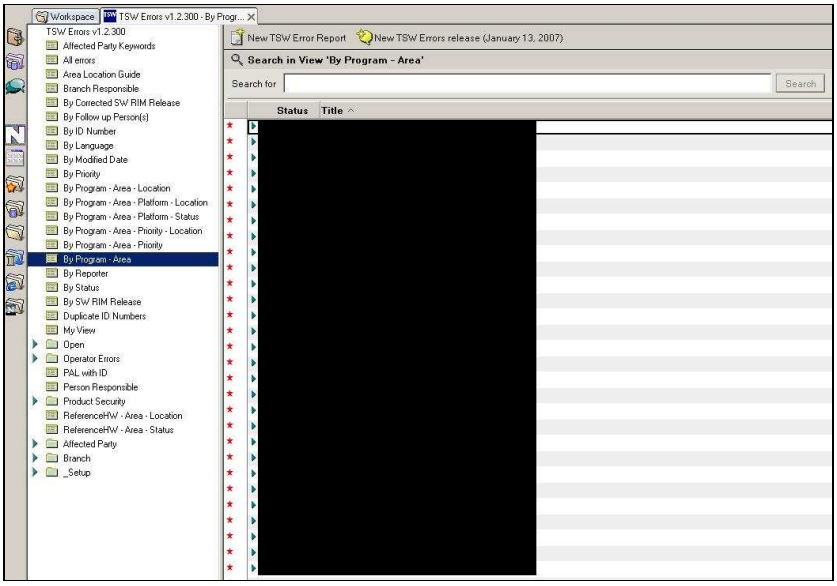

Figure 4 TSW view

4.3.4 PCP

PCP (Product Creation Process) database is similarly part of *Lotus Notes* system and it works same way as TSW. Only error focus areas are slightly different due organization

using it. It is mostly used for managing network and hardware related errors. The figure 5 is showing a sample view from PCP database. The actual content related to Nokia internal information cannot be shown.

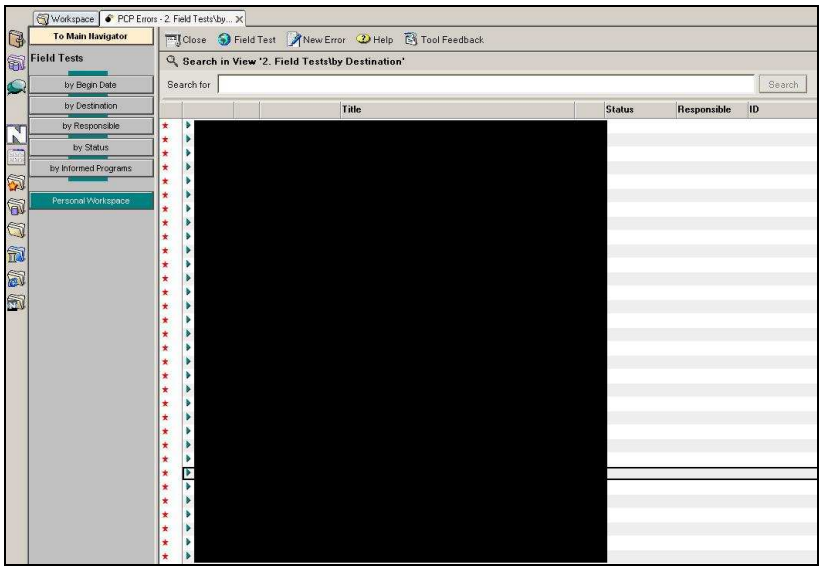

Figure 5 PCP view

4.3.5 Database Mapping

Database mapping is a vital functionality for managing and synchronizing errors between different databases. Status mapping is happening in three levels; R&D databases which includes PCP and TSW debases Internal OTC TRUE (Project TRUE) database and external OTC TRUE database. R&D statuses are mapping to internal OTC TRUE database and from there statuses are mapped to customers in external OTC TRUE database. Customers are only seeing the external OTC TRUE database information. Project TRUE, TSW and PCP databases views are only for Nokia internal personnel (Figure 6).

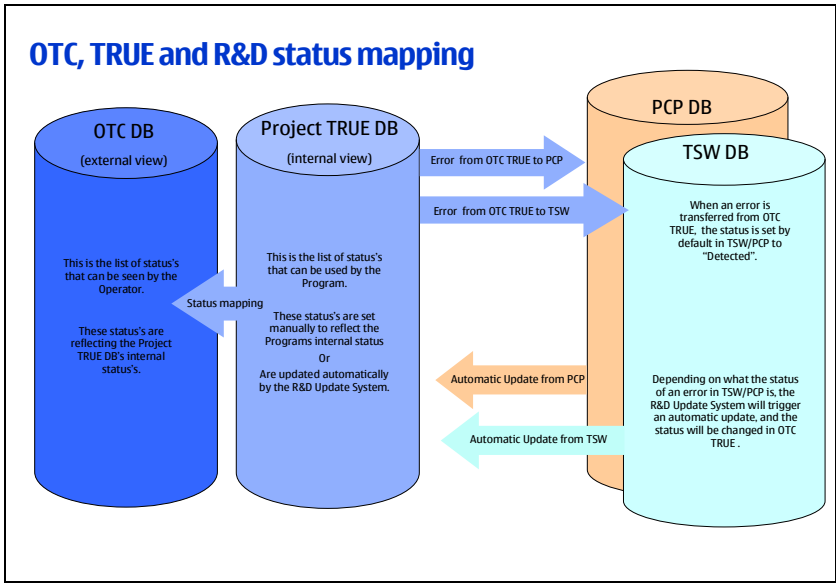

Figure 6 Status mapping between databases

The purpose of mapping is to make the most common status changes automatically between project OTC TRUE and R&D databases. On the other hand, mapping between external and internal OTC TRUE is fully automatic. This is one functionality which is reducing work in the issue handling.

There are certain status categories in R&D DB, which are indicating relevant situation for Nokia error correction process. One of the tasks in issue handling is to inform CTA and customers about status of error correction process. When this happens automatically, it saves work steps in issue handling.

Status is only giving indication about the error correction process. It does not give detailed information about the error and possible solutions for it. As well as OTC TRUE error reports are giving detailed description about customer problem, exact information about the error investigation and possible corrections are included in the actual R&D error reports.

5 Software testing

5.1 White box and black box testing

Testing is trying to validate functionalities of the software against requirements and find possible errors from the product. It is impossible to find all the errors but with a systematic and good test planning the most of the repeated errors can be found. Testing can be *static* or *dynamic*. Static testing means that the code itself will be analyzed manually or automatically. Desk checking is ensuring that the code algorithm is mostly working as it was expected.

Dynamic testing can be divided roughly in to the two separate parts; *white box* and *black box* testing. White box testing looks implementation as details. It will cover widely the whole structure and logic, since it is visible for a tester (Figure 7).

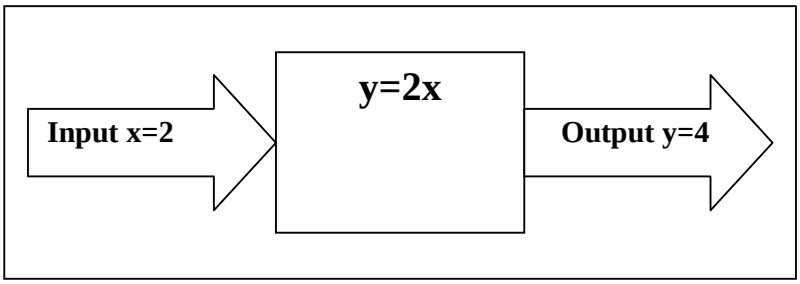

Figure 7 White box testing

In black box testing the internal implementation is hidden from a tester and testing is only focusing analyzing the results (output) of the software run test cases (input). In other words, in black box model, it is tested functional and behavioral issues (Tamres 2002: 220). Nokia acceptance testing is black box testing (Figure 8).

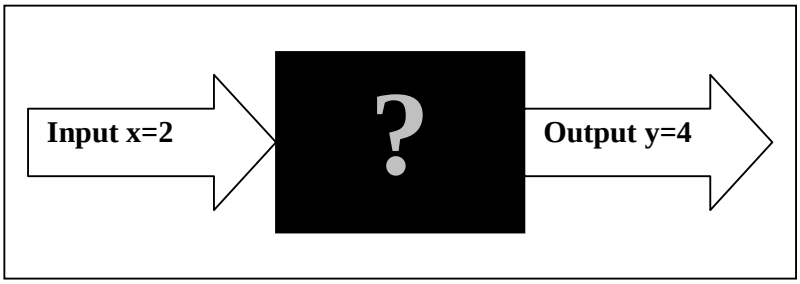

Figure 8 Black box testing

5.2 Amount of testing

Testing is good to be done thoroughly, but perfection is not usually cost-effective. From business point of view, it is worth to stop when the certain amount of testing is done. Optimal situation does not mean that tested object is missing quality, but if it is not question of costs, over testing will lower the amount of errors. Patton (Patton2001) presents the relationship between the amount of testing performed and the number of bugs found in the figure 9.

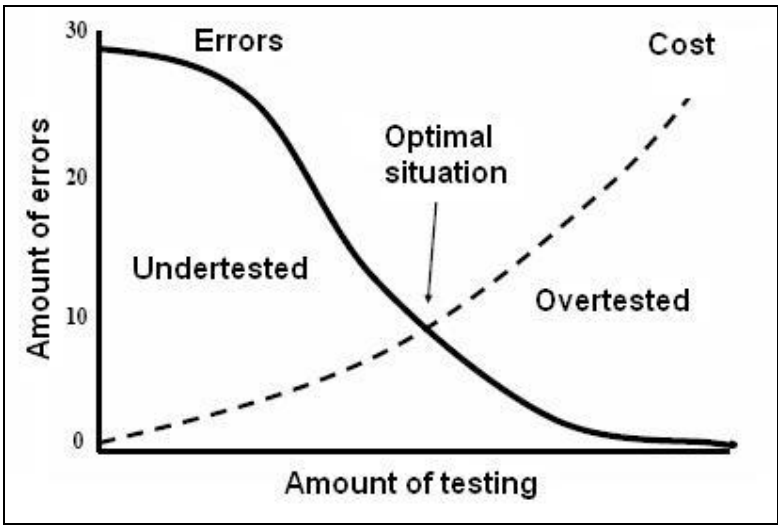

Figure 9 Optimal situation for testing (Patton 2001)

5.3 Concept of an error

Several synonyms for an error in software engineering are used. Defect, failure, fault or bug are widely used when talking about errors. Error refers to an incorrect action or calculation performed by software. When talking about testing, it is departure from specification (Haikala&Märijärvi 2004: 287). Error cannot exist without specification, because error is validated against the specification.

In general, an error results from a combination of a defect (code that does not correctly implement the requirements or intended behavior) and a fault. If, as a result of the error, the system performs an undesired action or fails to perform a desired action, then this is referred as a failure (Haikala&Märijärvi 2004: 288).

In the beginning of the product acceptance testing, it is normal that the amount of found errors stays high, compared to the fixed errors. Just after a while, the amount of fixed errors will reach the same level with the found errors, until all valid errors for product acceptance are corrected (Figure 10).

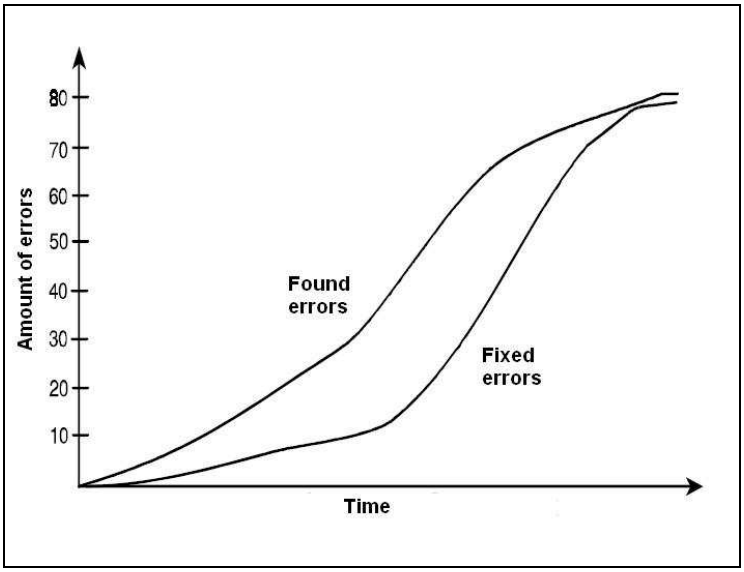

Figure 10 Error curve and fixed errors (Haikala&Märijärvi 2004: 293)

5.4 Error versus issue

In Nokia CTA terminology, an error is rather called as an issue than an error. The most relevant difference between an issue and an error is that, an issue, is yet unidentified problem whereas an error is proved to be caused by a fault. Reported customer issue is not always an actual error, but a feature which is working according to the specifications. Customers might report about issues what they see as an error but company might see that as a feature (Haikala&Märijärvi 2004: 287).

5.5 Error handling and error management

Usually, the main reporters are software testers and developers but it is not uncommon to receive reports from marketers, support stuff and customers (Bays 1999: 51). During the Acceptance testing, all errors are coming from customers or testers on behalf of customer. However, even if usual case is that most of the errors are already found internally, acceptance testing helps to find errors better from live customer environment. That is the reason why large amount of error reports are concerning on customer features and functionalities in the software.

Founded errors should be reported and analyzed. Not only that reports are the most relevant issue for fixing errors in R&D, but reports will remain as a document which can be used as a history when planning a new project. In Acceptance testing, a certain form to report errors is used. Thorough made form will force reporters to write down all relevant information which was related to the error. Report has to contain e.g. clear error description, expected and actual result. Clear error report will help developers to understand the problem easier and error corrections can be made faster.

Error handling and error management is relevant especially when the amount of errors is large. In error handling, the customer reports are checked and filtered by error handlers. At this point, invalid errors can be separated and valid errors can be delivered for error correction process. Purpose of error management is to control and monitor that correct error processes are followed and valid errors will be corrected. Error management, on the other hand, is prioritizing most critical errors for developers, so that error will be fixed in a correct order.

Classification can be used for making prioritizing easier. Errors can be categorized for different levels. Thomas Pyzded has used three different levels to categorize defects, which are presented in table 1(Pyzdek 2003: 218). In software engineering, the same categories defining defects can be used, but definition is needed to match software engineering concepts. Nokia is using the same categories for prioritizing errors. In addition to these categories, Nokia is using category *Showstopper* for defining extremely critical errors, which need immediate actions from error correction point of view.

Table 1 Pyzdek defect categories (Pyzdek 2003: 218)

Critical defect	A critical is a defect that judgment and experience indicate would result in hazardous or unsafe conditions for individuals using, maintaining, or depending upon the product or a defect that judgment and experience indicate is likely

	to prevent performance of the tactical function of a major end item such a ship, aircraft, tank, missile, or space vehicle.
Major defect	A major defect is a defect, other than critical, that is likely to result in failure or to reduce materially the usability of the unit of product for its intended purpose.
Minor defect	A minor defect is a defect that is not likely to reduce materially the usability of the unit of product for its intended purpose or is a departure from established standards having little bearing on the effective use or operation of the unit.

5.6 Error management versus issue management

In the same way as errors and issues, Nokia CTA error management is rather called currently as CTA issue management. However, CTA still continues to make error management work as well. When issue management exists, it comes always before error management, and it is emphasizing more about pre-analyzing issues. Issue and error managements are basically doing the same tasks, but in different level. Issue management is a front end work whereas error management continues from the point where issue management has ended.

5.7 Test planning

Project management is the main responsible for test planning. There have to be also responsible project management in customer side, in order to care of the test planning on their behalf, because Nokia cannot interfere customer's internal planning. Test planning has to include, at least, schedule estimation, staffing resources and equipment needs (Tamres 2002: 214). Tamres is also mentioning the following issues which, are relevant in a good test plan:

- Overview of schedule for test activities, which is used by project managers to produce the project schedule;
- Approach to testing, including usage of testing tools;
- Test tools, including how and when to obtain them;
- Process which conducts tests and report results;
- System test entry and exit criteria;
- Personnel required to design, develop, and execute tests;
- Equipment resources – what machines and test benches are necessary for;
- Test coverage goals, where appropriate;
- Special configurations of software and hardware needed for tests;
- Strategies for testing the application;
- Features that will and will not be tested;
- Risk plans.

5.8 Testing levels

The design for each testing level is done in the corresponding level where the actual testing is performed. Respectively, the results of each testing level are validated against the documents created in the corresponding design level (Figure 11).

Nokia is testing the software internally through organization levels by testing teams which have been set up by project management. In each level, there are responsible persons who create test cases for software testing. Ideally, it is a software tester who creates the test cases. E.g. platform software is tested using test cases created by a platform testing team and program software is tested using program testing team. In the end, software has to pass the test cases which are created according the specifications and required standards.

5.8.1 Module testing

Module testing is validating against software design and module planning documents. It is the first opportunity to test the created code in module level. Module testing can be done by the person who created the code, or it can be used separated testing personnel.

5.8.2 Integration testing

Integration testing is validating against architectural and system design documents. It is verifying that the combined modules work together correctly. Integration testing process starts with combining first individual modules and adding more of them until the entire application have been built. The testing is done in stages, making it easier to find the root cause. When error occurs, it is more obvious that the last added component will be the cause for the problem (Tamres 2002: 221).

5.8.3 System testing

System testing is validating the entire product against original project requirements and its purpose is to verify that all software and hardware components are integrated correctly together. Tamres mention following testing categories, which are included in system testing (Tamres 2002: 222):

Compatibility testing ensures that product will interoperate with other components and devices. E.g. Nokia phones have to work together with different kind of computers.

Configuration testing is using different software and hardware configurations in order to verify combinations that the system must support.

Functional testing is validating the requirements the system have to fulfill.

Install testing. It means testing various installation and un-installation procedures which have to work properly in different kinds of situations.

Load testing is trying to simulate real world interactions. System or application is under a range of loads, in order to determine at what point the system's response time decreases or fails.

Performance testing is ensuring that the system is responding within the specified time constraints. This is very important, especially in web applications.

Recovery testing. The device under test or a system is verified that it will recover to a useable state after having experienced a crash, hardware failure, or similarly damaging problem.

Reliability testing is ensuring that the system operates under certain conditions for a specific period of time.

Security testing is verifying that only authorized users have access to allowable features.

Serviceability testing. This is related to internal maintenance information, such as traces and diagnostics messages. It's also known as maintainability testing.

Stress testing is designed to confront abnormal situations and, its essentially purpose, is to break the program.

Usability testing measures how well people can use some human-made object for its intended purpose, e.g. usability testing measures the usability of the object.

5.8.4 Regression testing

When the changes are done into software which has been already tested, it has to be tested again. The purpose of regression testing is to ensure that changes have not broken any other part in the software code, meaning that a new fault is not found. The part changed will be tested with the old test cases, but also with new test cases, which are validating the error correction. After this integration, the testing phase will be repeated again.

5.8.5 Acceptance testing

Acceptance testing, as a concept, is quite unknown and it is often associated to release testing, usability testing, regression testing or alpha and beta testing (Haikala&Märijärvi 2004: 288-291). According to Ebelin, acceptance testing or qualification testing is supposed to demonstrate that the system design meets performance and reliability requirements under specified operating and environmental conditions (Ebelin 1997: 315). Acceptance testing is taking the place, when needed, at the end of the V-model. V model was developed in Germany in order to regulate the software development process (Figure 11).

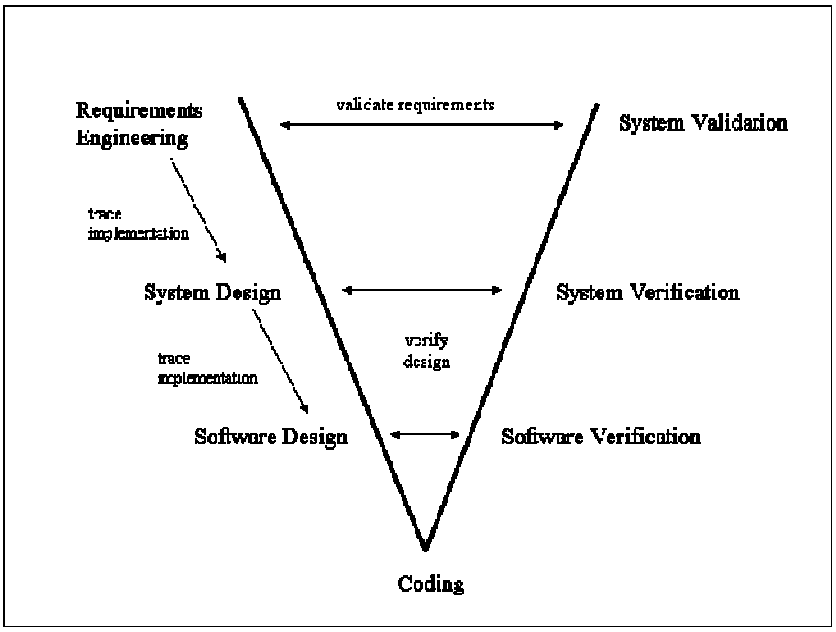

Figure 11 V-model

Acceptance testing is validating the system against customer requirements and ensures that the product is ready for operational use (Tamres 2002: 223). Depending on the software development area, the exact location for acceptance testing might be different. Acceptance testing can be both seen as part of the system testing or it can also come after it.

Acceptance testing can include or precede a process called *alpha* and *beta* testing. *The alpha* test is conducted at the developer's site by a customer. *The beta* test is conducted at one or more customer sites by the end user of the software. (Pressman 2000: 483.) The purpose of this process is to find errors that could not be found from internal testing.

Before starting acceptance testing, software maturity has to be good enough. In acceptance testing, the main goal, from corporate perspective, is to achieve customer's product acceptance as fast as possible, which does not necessary mean that all software related errors would need to be fixed by that time. Developing time will be very long if software needs to be 100% perfect. Finally, this depends on the product itself and the requirements what the customer has set.

Steve McDonald(McDonald 2000: 69) is giving some suggestive information about the relation between software faults and the time of developing the product (Figure 12). There is a point, when fixing more than 95% of the errors, delays the plans. Roughly it can be said that, acceptance testing is better to be started a little bit earlier when the amount of corrected faults are reaching 95%. That high fault correction number tells that software is enough mature, and most customer acceptance will obviously happen in a short period of time, because they would not find errors anymore.

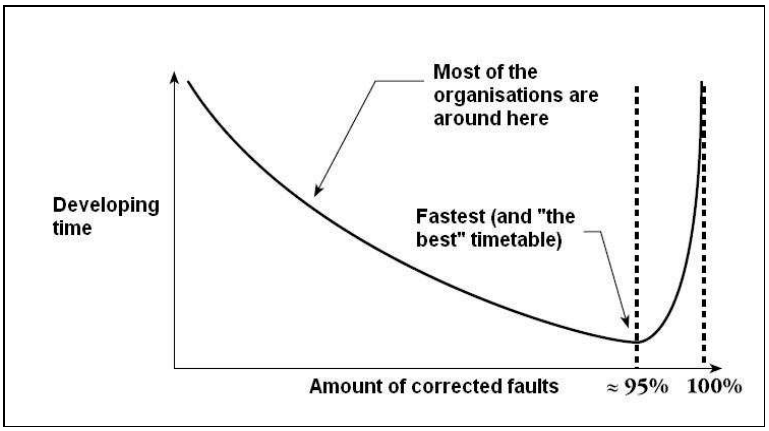

Figure 12 Optimized timetable

The figure does not show anything about the severity and priority of the software errors. It does not show about run test cases either. Figures do not represent the right fault correction status, if the test specification were wrongly designed, so real errors are not actually found at all. Practically, it is very difficult to make realistic plans for acceptance testing.

Optimized situation, for product acceptance testing, is that software would be mature enough, so that customers could run the test cases during *alpha* testing. In *beta* testing, all reported errors found from first round could be fixed in time, so customer does not find any more errors, which could prevent accepting the product. From mobile business point of view, it is also important to let customers test the product early enough during acceptance testing. Acceptance testing is also the opportunity for customers to tune up mobiles with the network environment.

6 Nokia acceptance testing

6.1 General

In Nokia terms, *alpha* is referring to *technical trials* and *beta* is the same as *acceptance testing*. Acceptance testing will end up in the customer approvals. *Technical trials* will differ from *alpha testing* in a way that Nokia will provide proto devices and tested software to the customers, and the actual testing will happen at the customer site, instead of developer's site. *Acceptance testing* will also happen at the customer site (Figure 13).

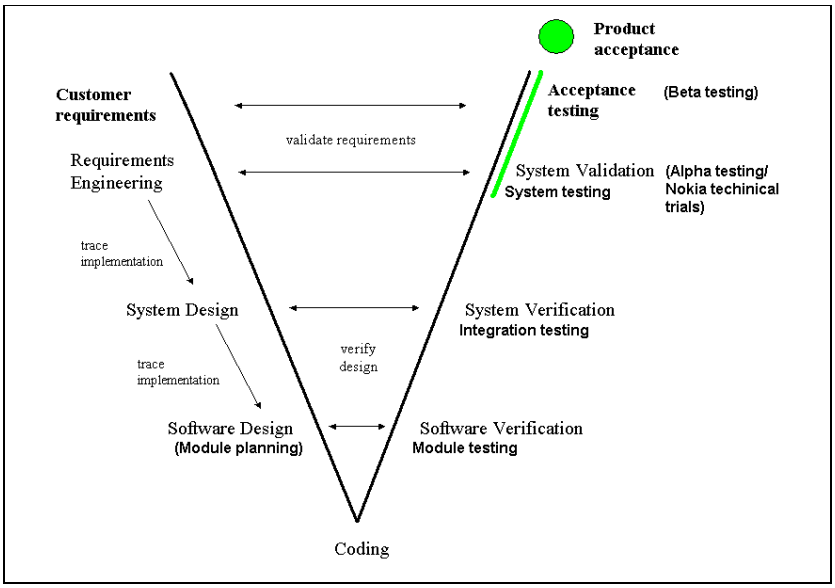

Figure 13 V-model with acceptance testing included

6.2 Test planning

Nokia acceptance testing is planned together with program, CTA and other relevant stakeholders. Planning includes estimation about resources, equipment, risks and schedules. Relevant information for acceptance testing is communicated and negotiated with customers after Nokia internal planning. The most important information for CTA and customers is the schedule and estimation when the acceptance testing can be started. For the customer, it is vital to know the schedule, so that resources can be reserved in time.

6.3 Starting acceptance testing

When software is mature enough and it fulfils all needed requirements, program will make the final decision when the software is given to the customer for testing, and when the actual acceptance testing can be started. All the necessary equipment and software is delivered to customer, who will start the testing. At the same time, issue and error management prepares to receive issue reports from the customer testers. Resources, tools and access rights have to be ready before the reporting starts, so that valid errors can be fed to the software error correction process.

6.4 Customer testing and reporting

Customers are doing the black box testing, once they are testing the hidden implementation of the Nokia product. The only thing what Nokia will know about customer testing is the result of the failed test cases, which are collected to the issue reports. Actual test cases and other details about the testing might not be visible, because customers are not always providing that information for Nokia.

Customers are writing the error reports according to their own processes. However, Nokia is providing guidance for customer testers to ensure adequate quality of issue reports. A good report should always include basic background information about the test report: who was testing and where, what was the product software version and what kind of environment was used for testing. E.g. which network was used and what possible other applications were used during the testing.

The most important thing is the exact issue/error description. The expected and actual results of the test case, testing steps which are leading to the error and how often the error was able to reproduce have to be explained. Also screenshots, error messages and error logs will help the investigation a lot. Without decent error description, Nokia cannot help customer to solve their problems.

6.5 CTA issue handling

Customers have an access to the Nokia OTC TRUE system, where they are inserting the issue reports. Nokia issue/error management filters and analyses the reports. Valid reports are transferred for further R&D investigations and the invalid reports will be rejected. Figure 14 is presenting an example of one OTC TRUE error report.

Figure 14 OTC TRUE DB error report.

6.6 CTA responsibilities

After valid errors have been transferred to R&D, CTA is responsible for following up the errors, which are possibly gating the customer product acceptance. This work is dedicated to CTA responsible persons, who are negotiating about the issues in different levels in the Nokia organization, having meetings where error corrections are discussed together with software responsible persons. In the meetings, responsible persons are reviewing the current status and decisions are made about error correction priorities, schedules and targets. If there are any high priority customer issues that needs to be solved, then CTA is responsible to inform software creation team about this kind of information.

CTA is also having similar meetings with bigger customers, like mega operators. In the meetings, the status of customer plan and error correction priorities is reviewed. CTA will take input from customer, discussed again with Nokia R&D. CTA needs to know also the current Nokia software status, so that information can be discussed together with customer. E.g. issues which Nokia has rejected from error correction plan have to be explained to the customer. It is the customer decision if explanations are acceptable or not.

6.7 Customer product acceptance

Product will be accepted when customers get convinced that the product is ready for market sales. Practically, approvals mean a situation when customers are not finding errors anymore, or when the severity of the remaining errors is so low that the product is good enough. After the approvals, there is a possibility to make an agreement where manufacturer promises to fix remaining errors for the customer. If the customer requires approvals for future maintenance releases, CTA also provides that.

7 Old CTA error handling

7.1 Old OTC TRUE status

The old OTC TRUE DB was consisted of 13 different statuses, mapped between internal OTC TRUE and external OTC TRUE databases. Customers were able to see a slightly difference in status from the external OTC TRUE view, since *R&D ignored* in internal view, mapped to *rejected* category in external view. In addition, certain R&D status categories were mapped to the OTC TRUE DB. The purpose of the error handling flow was to handle all the open errors within the certain time, and achieve final status in OTC TRUE DB.

Final status means that the error report has gone through the error handling process and has faced the final decision, which cannot be changed without good reason. The reason for changing the status might be e.g. new evidence has been found for an invalid error, which has turned to be valid error. *Final status* errors can be: *closed, solved, as specified, not an error, R&D ignored* (*rejected* in external view)

Here all the old OTC TRUE DB status categories from Nokia error management point of view are explained:

1. Open

New externally reported error from originator which is not handled yet. *Open* reports needs to be handle primary. This could also be a report which has been in *operator query* and has been returned back, to be *open* when originator has answered to the query. *Open* status is used in both internal and external views.

2. Operator Query

When error report is unclear, more information is possible to be asked from originator using *operator query* function. This function is only used when the handler expects clarification about the error from the originator. E.g. this function is not used when just commenting the error. *Operator query* status is used in both internal and external view.

3. Not an Error

A report is not seen as an error at all. The content of the report is not related to errors. E.g. a report includes only irrelevant questions or comments, or it is just a blank report. *Not an error* status is used in both internal and external view.

4. Known

An error has been reported before and the report can be already found from OTC TRUE databases or other R&D databases. There has to be a link to the duplicate error, or a *track known error* function is used which makes a link between the errors. Duplicate errors needs to be followed up because of customer updates in OTC TRUE database. Known status is used in both internal and external views.

5. As Specified

When error is working according the specifications. New requirements and customer suggestions are belonging to this category. *As specified* status is used in both internal and external views.

6. Report Accepted

The error has been accepted to be a valid error and it has been transferred to R&D database first time. *Report accepted* status can be used also for duplicate errors, if there is a wish to emphasize that particular customer error in R&D error correction process. In those situations, the link has to be created manually or with *track known error* functionality. Other option is to use *known* status. *Report accepted* status is used in both internal and external views.

7. Studied

The error is investigated by error handling team and not by R&D team. It is also used when the issue status is pending. *Studied* status is used in both internal and external views.

8. Solved Not Released

Correction is available, but release schedule from product program side is still open. *Solved not released* status is used in both internal and external views.

9. Solved

Error is solved: the corrections are available and error is verified. The error is not reproducible or issue is no longer reproducible, even though there is no exact fix for the problem. *Solved* status is used in both internal and external views.

10. R&D Ignored

An error report has been ignored by R&D. The status has been changed to *ignored* in R&D databases, which is mapping back to OTC TRUE database. This is seen as *rejected* in external view.

11. Rejected

The same meaning as *R&D Ignored* but this status is seen externally by customers.

12. R&D Duplicate

An error report has been set as duplicate in R&D database. *R&D Duplicate* status is used in both internal and external views.

13. Closed

Originator has closed the error. Open status is used in both internal and external views.

7.2 Process overall

Customers, who had the agreement with OTC, were reporting errors to the OTC TRUE DB. Error reports were seen as an error since the old process was not aware about the concept called “issue” and that was the reason why current issue handling was called before as error handling. In error handling, information was going between customer and the developer, including acting operator contact person and error handler as in the middle.

All open errors reported by the customer were handled through error handling team. Based on the available information in the error description, reports were accepted or rejected. If the report was having lack of information about the problem, it was sent back to originator and asked more information. Other OTC TRUE DB status were used based on the OTC TRUE error description and was set to the status, which was the most suitable for the situation.

Accepted reports were transferred to the R&D databases, where those were set under specialists' responsibility. Status of the error report after that was changed according to the information given in R&D databases.

7.3 Old mapping between databases

Status between external and internal OTC TRUE databases were correlating with each other, excluding status *R&D ignored*, which was shown on the external database as *Rejected*. When the originator answered to the *operator query*, it turned back to *open* again.

When errors were transferred to TSW DB, OTC TRUE status was changed to *Report accepted* and R&D status was changed automatically to *Detected*. When error status was changed to *Ignored* or *Postponed* in R&D DB, status was mapped to *R&D ignored* in OTC TRUE DB. Duplicated errors were mapped in OTC TRUE to *R&D duplicate*. *Verified* and *Closed* statuses in R&D DB were mapped to *Solved* in OTC TRUE DB (Figure 15).

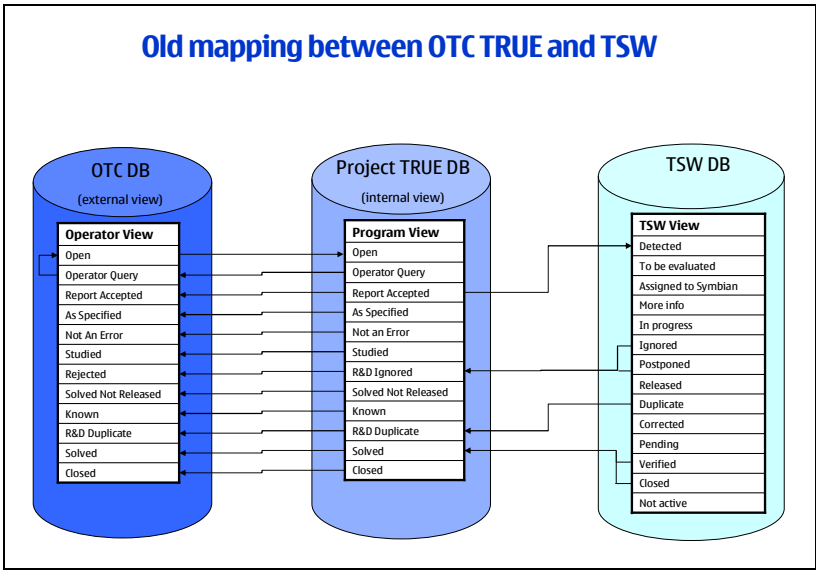

Figure 15 Old status and mapping between OTC TRUE and TSW

When errors were transferred to PCP DB, OTC TRUE status was changed to *Report accepted* and PCP status was changed automatically to *Detected*. When error status was changed to *Ignored* or *Postponed* in R&D DB, status was mapped to *R&D ignored* in OTC TRUE DB. Duplicated errors were mapped in OTC TRUE to *R&D duplicate*. *Verified* and *Closed* statuses in R&D DB were mapped to *Solved* in OTC TRUE DB (Figure 16).

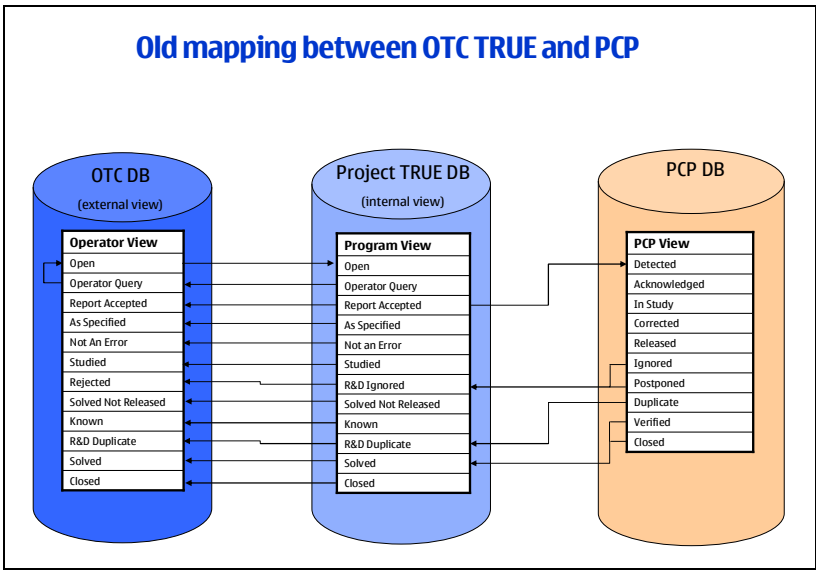

Figure 16 Status and mapping between OTC TRUE and PCP

8 Problems and solutions

8.1 Documentation in general

Most of the problems happened because of the lack of appropriate **documentation, instructions, communication and lack of adequate training**. This problem was seen both in customer's and in Nokia's side. There were not good instructions for originator to report the errors correctly for Nokia. Also, the people in the middle of these two parties did not always have sufficient technical competence to understand error reports. As a result, there were too much error reports fed into error database, which were not understood. The worst case was that errors which were not understood were not corrected at all. On the other hand, errors which were thought to be errors, ended up not to be errors, wasting R&D resources. That's why those were not even seen as real errors, causing delays in the correction process.

On the Nokia's side, there were not good simple guidance to handle CTA error reports and the understanding of the errors was insufficient. Error handling had problems sometimes to follow strictly the process, causing misunderstandings and software correction delays. However, even if Nokia R&D had good error management documentation and technical competence, R&D was not totally aware of the importance of the customer's errors and they were unconcerned about the CTA errors.

This problem is complex and needs several actions in different areas. First of all, there have to be proper documentation and instructions for error handling work, so that work will be done correctly and systematically. Training has to be increased, so that adequate competence level is fulfilled for people who work with issues and error reports. Visibility of CTA through organization streams have to be stronger when interacting with stakeholders. CTA have to communicate more closely with customers as well as with R&D, and take care of prioritized and relevant issues. Communication means also cooperation from the stakeholders, so that all parties are aware of situation and understand errors correctly.

8.2 Process

The biggest problem for CTA error handling process was the non-existence of the documentation. Work was done before like "ad hoc" style and even if there was a certain process for the error handling, it was not written down properly. If there is no clear existence for a process, it cannot be followed.

The solution for this is to identify the process for CTA issue handling and write it down. Process has to be exact enough, so that all phases and tasks can be explained. However, it shouldn't go too deep in to details, because process needs to be still generic and understandable. Generic process in to the right perspective gives space if something goes wrong, what was not taken into account when creating the process.

8.3 Status categories

OTC TRUE DB was having too much status categories and it was found that the categories needed updating. It was also wanted to make categories simpler for customers in the external OTC TRUE view. The reason for this was that customers were only interested in their issue corrections. If error was not corrected, there should to be a good explanation. Simple status category is not only useful for customers, but also for error management because it is making error handling more straightforward.

The solution for this is to remove all unnecessary status categories and use only categories which have a clear purpose in error handling work. Categories have to be also generic enough, so that status is holding it's meaning in several projects which are using OCT TRUE DB. Otherwise, there is a risk that categories are needed to be changed every now and then, which is just consuming resources.

8.4 Mapping

The purpose of mapping was making the error handing work easier. Mapping was affecting between internal OTC TRUE, external OTC TRUE and R&D databases, but it was not working perfectly. Mapping had to be changed between internal and external OTC TRUE because status categories changed. Mapping had also to be changed between internal OTC TRUE and R&D DB. Status categories in R&D DB did not always corresponded to the status in OTC TRUE after mapping and R&D feedback had to be explained in details anyway to the customer. Error handler had to separately, open R&D DB manually and check the information which had to be communicated to the customer. The action to open the error report in Lotus Notes was only adding extra step, and it might take sometimes a lot of time depending on the load of the system.

Another problem in R&D database was that mapping did only work with *common base status,* which was used generic by *S60* in software correction process. However, CTA was following the software correction status of *program integration team,* because it was the department after all that was combining the software for customers. Now, the system did not map R&D program integration status to the OTC TRUE DB at all.

As a solution mapping have to change to match status between internal, external OTC TRUE and R&D databases according to the new categories. Mapping has to be as simple as possible to reduce the manual work of error management, so that the need of opening the R&D DB could be minimized. This is because it is very common that working with Lotus Notes databases, gets slower when the load of the system will increase, or the amount of data in databases grows too big. Mapping has to be activated when *program integration status* is changed instead the *common base status*.

8.5 Other problems

In addition to the previous, a group of separate problems were identified, which were affecting the issue management work negatively. Several times problems were found in practical OTC TRUE database usage, caused by replication, data losses and unwanted breaks during work day. Information was missing and data wasn't updated. Sometimes problems were caused just because of development and support work in the OTC TRUE system. Some of the problems were related the "nature" of Lotus Notes system, such as replication and data conflict issues.

There is no easy solution for replication and data losses, once those are issues which have to accept when using Lotus Notes system. In addition to that, it's more like a technical issue what system support has to solve. System support, however, could help with this issue, giving guidance and information about their development schedules.

Another big problem was the CTA *region* and *country* issues, because those did not have account and location in OTC TRUE DB. Issues were reported by email, which made it difficult to follow and track them systematically. A solution for this was to ask support team to open an account and create a location in CTA section in OTC TRUE DB.

It was found that OTC TRUE system was missing tools which could help managing errors get better. There was a need to import, sort and export outside of the OTC TRUE system, because the system itself is very heavy and slow to use for just reviewing the errors. Also, there was a problem to get trustful statistic for management use. For some reason, system created duplicate error items in the database, which distorted the actual results. A tool for import/export functionality has to be built and also the way to get correct error report statistics for CTA purposes.

Although, Nokia was providing guiding and ready-made form for people who reported the errors, instructions were not always followed. Customer error reporters were occasionally creating incomplete error reports, which made the investigation more difficult and in a worst case, the error could not be investigated at all. As a solution, there is a need to improve the guidance and inform all the people who were creating customer error reports.

9 Improvements in CTA issue handling

9.1 General

Improvements of the documentation were concentrated for the CTA issue management. The purpose was to write down what the issue management work was all about. The meaning of the documentation is not only giving guidance for people who are already dealing with CTA issues, but also new personnel for training purposes. The idea of process description, work flow and explaining tasks in details were to ensure that work in CTA team was done correctly and more efficiency. All new relevant CTA documents are introduced in this chapter.

Another big improvement was changes in status categories and new database mapping. Changes were done in the OTC TRUE system in autumn 2006. Descriptions for new status categories and new database mapping are found in this chapter as well. Actions for rest of the improvement have already started and plans are in progress.

9.2 CTA issue flow

CTA issue flow was documented in high level to describe how the issues are moving through Nokia software error correction processes and returning back to the customer. As an input, the issue report was given with a certain description and as an output possible software fix or rejection with explanations. Explanation is corresponding to the number mentioned in the figure 17.

1. Originator creates an issue report as an input and inserts it to the OTC TRUE system.

2. CTA issue handling team is receiving the issue report and issue handling is done using OTC TRUE database. Issues are handled according to the CTA issue handling process and valid errors are transferred for *program* in R&D databases. Invalid errors are rejected and communicated back to the originator.

3. *The program* is investigating the errors and following their own error handling process. If the errors are solved in *program* level, then it will be communicated to CTA issue handling team, which is delivering the information for originators.

4. If the errors need to be solved in *platform* level, then *program* will forward the errors to *platform* responsible area. Also, CTA issue handling team is allowed to transfer the errors directly for *platform*, if the error is proved to be a *platform* related issue. Errors will go through the *platform* error handling process and when errors are solved in *platform* level, information is communicated to back to each level, *program*, CTA and originator.

5. If the originator is happy about solving the problem, the issue will be closed.

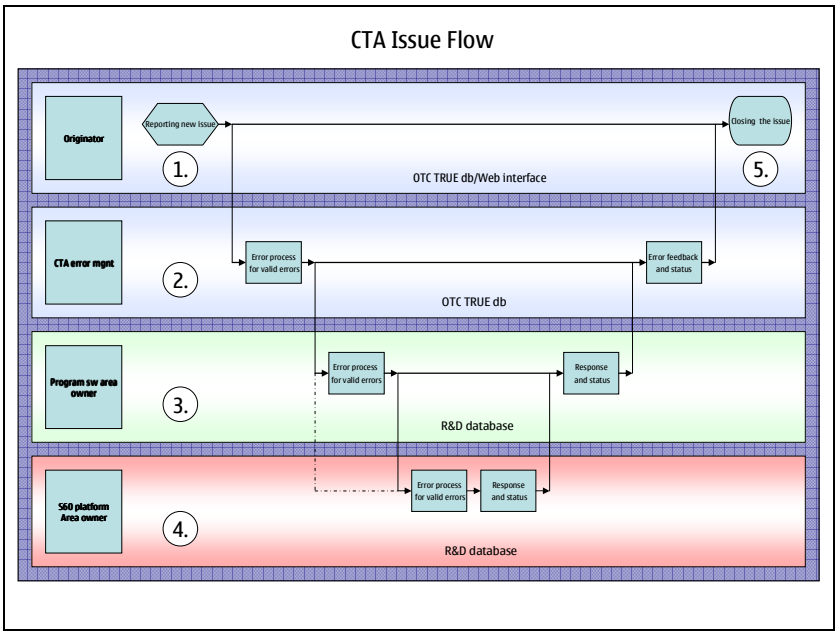

Figure 17 CTA issue flow

9.3 Improved CTA issue handling process

The whole CTA issue handling process is divided in steps in the table 3. Each step is explained in description column and there is corresponding index number in Figure 18 matching to the index numbers in the table 2. This is the most relevant guide for concrete CTA issue handling.

Table 2 CTA issue handling flow

Index	Description
1	Originator (operator) finds an issue in a product and reports a new issue report.
2	The issue report is visible to the issue handlers in OTC TRUE database as a new *open* issue.
3	The issue is prioritized by issue handler. The most high priority issues are handled first.
4	The issue needs to be understood. More information can be asked from originator by using OTC TRUE system function *operator query* which changes the issue status and sends questions back to the issue reporter. The reporter will get an email where the issue report link is attached. The link will take the reporter to the issue report where he/she can see the questions.
5	When the originator has received and answered to the query, the issue report will come back as an *open* issue report with the added operator comments. *Operator query* function can be used as long as it is needed, until the issue handler has understood the issue report.

6	At this point, valid and invalid errors are separated. The issue is checked against the test case and the current test specifications. If the issue is working according to the specifications, the report will be rejected with explanations. The report is also rejected if the content of the report has nothing to do with the current testing. Rejected reports need to be commented carefully, because customer doesn't always understand why their issues are rejected. If there is not good explanation for the rejection the customer might demand to solve their problems, and in the worst case, they will not approve the product, and no sales is going to happen.
7	All valid errors need to be checked if there are already duplicate errors in the R&D database. If this is not done, it is taking extra time for R&D to investigate duplicates errors instead of investigating and fixing relevant valid errors. Always when the duplicated error is found, it is has to be linked to the OTC TRUE error report and vice versa. In that way errors, can be tracked down later.
8	If the issue is reproducible, it can be transferred to R&D for further investigation. Before transferring issues to R&D databases, it is recommended that issues are reproduced by the issue handlers with the same testing environment that the customer has been using when they have found the issues. Issue handlers need to follow the same testing steps as the customer has used in the test case. Also, exactly the same software version needs to be used. If the issue is reproducible, issue is validated and the root cause can be found. This kind of evidence is very relevant for R&D developers, when they are trying to investigate the actual error. The issue will be transferred to the S60 area owner if the issue has been reproduced with S60 software. If the issue has been reproduced with the program software, it will be transferred to the program area owner. If the issue is not reproducible, then handler will reject the issue and comments that to the customer through the OTC TRUE database.
9	Issues need to be reproduced and verified on newer software version. This is done because issue/error might be already fixed on the newer software version. If the issues/errors are verified already at this phase it will save a lot of R&D resources and it will possibly shorten the product approval time. If it is proved that the issue is not reproducible on the newer software version, it will be communicated back to cus-

	tomer through the OTC TRUE database. Customer is also informed about the software version which the issue was not reproducible anymore.
10	All valid and non-duplicate error is transferred to R&D databases. For every product and function area there is a certain location in R&D databases, where the errors are transferred. For an every location area, there are assigned at least one person to take a responsibility for an error. Responsible persons need to take care that the errors are handled according to the error management process.
11	Customer wants information about their error corrections. That's why errors are needs to follow up all the time and communicated about the possible error corrections back to the customer. CTA's operator responsible persons are following the sales gating issues regularly, but this is done also by the issue handlers. The developers, who are investigating the errors in R&D database, need sometimes more information about the errors as well. Issue handlers are communicating closely with developers in the R&D databases and providing more updated information related the error. In addition to the normal information, issue handlers are delivering developers different kinds of test content and other evidence provided by customers. When issue handler needs to provide more information for developers, it is done similarly as in step number 4 (*operator query*).
12	If an error is corrected, it will be communicated to the customer. It is also recommended that issue handlers will verify the error correction before it is communicated to the customer. If the error is still reproducible, then it is communicated back to the R&D and it will be asked to fix the error again. Due to the lack of time to verify all errors, customer can also be requested to retest with the corrected software. This is not however recommended, since without internal verification it is unsure if the correction will fix the customer error.
13	All errors which will end up to the final status are agreed with the customer, if those can be closed. Errors statuses which the customer will not agree will be negotiated between Nokia and customer.
14	Issue will be closed.

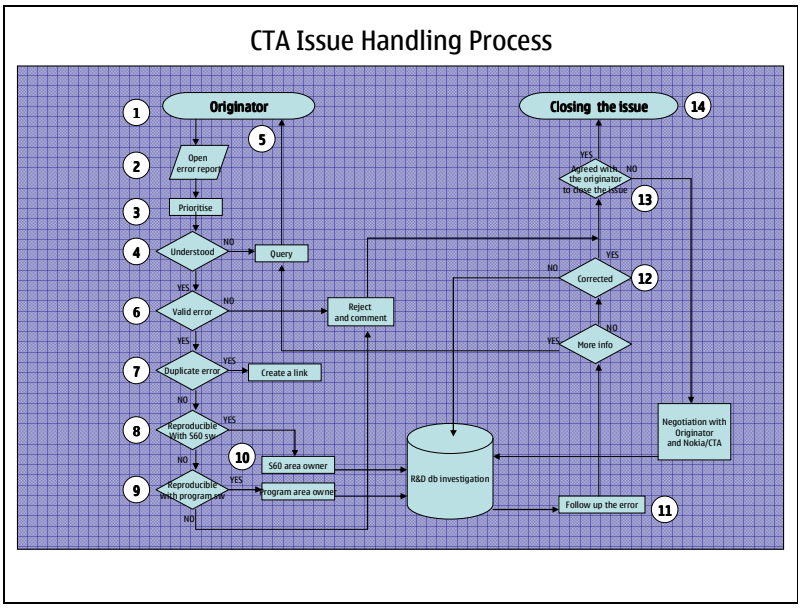

Figure 18 CTA issue handling process

9.4 Tasks identified in CTA issue handling

CTA issue handling process can be divided to smaller parts, which helps to understand issue handling details better. CTA issue handling process includes different kinds of tasks which are needed to perform, so that the process will be as complete as possible. An essential part of CTA issue handling is the filtering and pre-screening of the issues. Also, issue analysis is a relevant part of the process. There are 6 different main sections in the process. Tasks and descriptions are explained in the table 3 and corresponding numbers can be found from figure 19.

Table 3 Tasks involved in CTA issue handling

1	**New Open error report**
1.1	Customer is reporting a new issue report in OTC TRUE database.
2	**Prioritization**
2.1	Checking the most important issues first. Each issue handler has to recognize the critical issues which are going to gate the sales.
2.2	Sorting the issues by the severity. The operator is prioritizing the issues marking them, based on severity. There are 3 different severity levels in OTC TRUE DB indicating the importance of the issue for the originators: *fatal*, *severe*, and *minor*. Fatal issues are usually sales stoppers and those are needed to be corrected before the agreed approval deadline. Minor issues are not necessarily expected to be corrected. However, the issue severity can be still upgraded afterwards by the operator. In that case, it means that the minor issue is going to be upgraded to at least severe issue and it needs more serious actions than minor one.
2.3	Sorting the issues by the originator. Operators are prioritized

	based on how big business partner it is going to be for Nokia. High priority operator issues are going to be handled first.
2.4	Sorting the issues by the focus testing area. There might be a certain testing area that needs special focus and those issues are handled first.
3	**Investigation**
3.1	Understanding the issues. Issue reports need to be read thoroughly and the actual issue has to be understood before the issue can be forwarded to the *program* or *platform* as a possible error. If the issue report is not understood correctly, solving the issue is going to be difficult. If the issue is wrongly understood, the resources are bound to investigate totally different issue that was reported by a customer. In a worst case, the actual issue/error is not going to be fixed, because the information has been misleading and the assumed issue/error is not reproducible by R&D. In addition to that, wrong understood issues/errors are wasting R&D resources and the actual issue might still be a problem for a customer.
3.2	Studying the issue. Since operator is testing a large scope of test cases, there are various complex errors reported in the database. Issue handlers need to have wide technical knowledge to understand the root cause of the error. To fulfill lack of knowledge, the issue handler needs to find the source where to collect more information about the technical issue. In that sense, the issue handling is constant learning.
3.3	Finding duplicates. Duplicate errors are errors which have the same root cause, in other words, the errors which are the same between each others. Duplicates errors need to be eliminated, so that unnecessary resources are not used to investigate them.
3.4	Reproducing on valid software. There have to be evidence that the errors are valid ones. Reproducibility validates the error. After reproducing the issue it is easier also to understand the test case what the originator has used for testing. Based on reproduction, some conclusions can be made if e.g. the tester has tested against the correct specification, or if the test case itself is incorrect. Reproduction need to be done exactly on the same software version as the original issue. However, the issue is good to reproduce also with a newer software version, because if the issue is not reproducible with a newer software version, then it can be concluded that the issue/error has been already fixed.
4	**Action**
4.1	Transferring valid errors. Valid error will be transferred to R&D

	database. There are mainly two different databases where the errors are investigated: TSW DB and PCP DB. There is an instruction in OTC TRUE documentation, TSW, PCP and *programs error management* guides how the errors are transferred and what additional details are needed in error reports.
4.2	Finding the correct area owner. When the error has been transferred to R&D databases, the correct area owner needs to be assigned for the error. Certain developers are assigned to every area, which are then investigating and taking responsibility of the error correction.
4.3	Sorting out invalid errors. All invalid errors are rejected and commented back to originators.
4.4	Linking the valid duplicates. The reported duplicate errors are linked to the errors, which are already found.
4.5	Making comments and queries. If the issue report is not understood, the OTC TRUE system allows making a query back to reporter to ask more information related to the issue. It is also important always to comment the error when there has happened changes to the error during the error handling process. A meaning of the comments is to give up to date information about the error correction to the customers.
5	**Follow up the error correction**
5.1	Follow up is done in the same way as the tasks mentioned in the prioritization section. Customer needs information about the error corrections and issue handler has to follow up the status in R&D databases.
5.2	Updating the statuses. The statuses are giving the indication about the error corrections for a customer. The changed error status needs to be communicated to the customer.
5.3	Updating the comments. Status is not always giving complete information about the error corrections and detailed information needs to be provided to the customer by commenting the error e.g. what is the corrected software version. Communication is used to fulfill the missing information. The way of communication needs to follow the appropriate CTA process.
5.4	Verifying the errors. Error verification is not under CTA team responsibility, but it is done because the information about the error correction is delivered faster to the customer, and it will also fasten the customer approvals. The reported error is reproduced with the corrected software version, and if the error is not reproducible, then it is verified.
6	**Closing the error**

6.1	If the errors are verified and solved, and it has been agreed with customers, then the error will archived the final status and it will be closed. There might be cases where the final status need to be negotiated with the customer, e.g. if the error is working according to specification but the customer does not accept that. There also might be too big risk to fix the error without making the software quality go worse, which is also called regression.

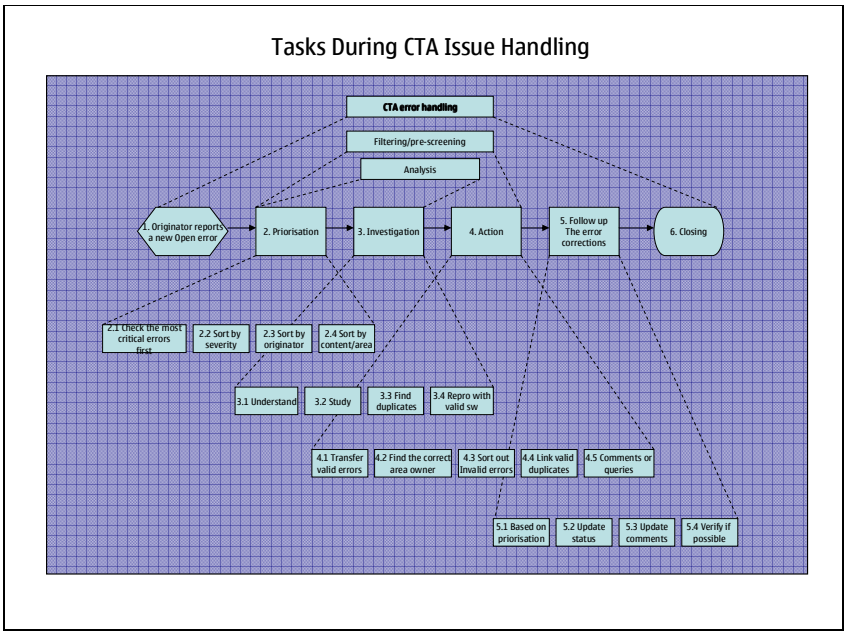

Figure 19 Tasks during CTA issue handling

9.5 Improved status categories

Current OTC TRUE DB contains less status categories than before, which is making the issue handling simple. Two new statuses were added in the system. Purpose of *investigation ongoing* status is to give positive indication for the customers, that their issues are in Nokia error correction process. *Action needed* status is added in the system to make issue handling more straightforward and it can be seen as a basket which collecting all the reports needed to be handled in one place. Before, status from R&D DB was mapped to several different OTC TRUE statuses. Currently mapped errors are collected under the one status, *Action needed*.

Five status categories wanted to be removed from the system. *Not an error* and *Rejected* were giving negative expression for customers. *R&D ignored*, *R&D duplicate* and *Duplicate* were seen useless and unpractical for issue handling. Those were mapped to OTC TRUE DB when error reports were identified to be ignored or duplicate errors in R&D DB. Those status categories were only temporary pending reports until issue handler had to change status to one of the final categories (*solved, closed, as specified, solved not released*).

New status categories means also changes to the final status. Final status is meaning the last step for an issue report. After the final status, error report will not be followed

unless it is opened again for some reason. Final status is including the following categories: *closed, solved, as specified*. New status categories are described in the table 4.

Table 4 New OTC TRUE Status and description

Status	Description
Open	New externally reported error (from originator) which is not handled yet. These are primary needed to handle.
Operator Query	When error report is unclear, then more info is possible to ask from originator using *operator query* function. This function is only used when the handler expects clarification about the error from the originator. E.g. this function is not used when just commenting the error.
Report Accepted	Error has been transferred to TSW/PCP and it is under R&D investigation.
Investigation Ongoing (New Status)	When status is changed to *report accepted* in Project TRUE DB, status is mapped as *investigation ongoing* to the OTC TRUE (Customer side).
Studied	The error is investigated by error handling team and not by R&D team. This is also used when the error is pending.
Known	Error has been reported before by any operator or duplicate error can be found from TSW DB. There has to be a link to duplicate error or *track known error* function is used. Duplicate errors are needed to follow.
As Specified	When error is working according to the specifications. New requirements and invalid error reports are also put under here.
Solved Not Released	Correction is available, but release schedule is open.
Solved	Error is solved: the corrections are available and the issue/error is not reproducible or issue/error is no longer reproducible even though there is no exact fix for the problem.
Closed	Originator has closed the issue/error.
Action Needed(New status)	Automatic Update returns the error to *action needed* when status in TSW/PCP is one of these: *more info, ignored, postponed, released, duplicate, verified, closed*. Error handler will set

	a new status in OTC TRUE based, what is the status and comments in TSW/PCP.

9.6 New logic for setting the status

Before proper documentation, it was not always clear how to use different status categories, even if there were definitions for each status. Even if it is difficult to find correct status for complex issues, documentation guides to choose suitable status in most common situations. In the figure 20, logic for using status categories in OTC TRUE database is explained. The most common cases are numbered and corresponding number can be found from a figure 20. Red lines are meaning the optimal way to handle the issues; blue ones are representing the optional way.

1. Originator creates a new issue report, which is set to *open*. Issues are prioritized and understood. If the report is unclear and it needs more information, issue handler creates a query and status is changed to *operator query*. After originator answer to the query, report will come back with *open* status.

2. If the issue is known or if there is already duplicate error created, report is set to *known*. At the same time link is created with known issues or duplicate error reports. Originator will see the same report as *investigation ongoing*.

3. All issues, which are working according the specifications, are set to *as specified,* which is seen in originator side with the same status.

4. All valid and reproducible issues are transferred to R&D database, where the correct area owner is located to be as responsible person for the issue. Originator will see the status *investigation ongoing* in the external view whereas *report accepted* is shown in the internal view.

5. Issues which need more internal investigation have to move under *studied* status. Originator will see the status *investigation ongoing*.

6. Errors which have a correction solution, but it has not been implemented to the product software, are set with the status *solved not released,* which is seen with the same status for originator. If an error will be not corrected, there have to have a negotiation with customers. Negotiations are out of range, from issue handling perspective.

7. Certain status categories in R&D database will activate the mapping functionality, which will change status to *action needed* in OTC TRUE. Since this status is working automatically, there is no need set it manually. However, reports under that category have to be handle manually and set with the new correct status.

8. Verified issues and errors are set with *solved* status and seen with the same status for originator.

9. After originator has agreed that the issue is finally solved, a report can be set to *closed.*

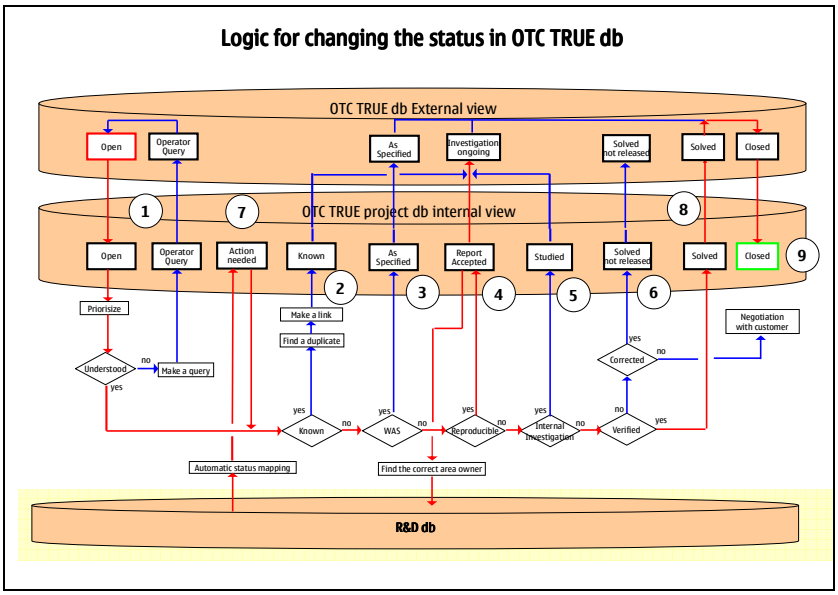

Figure 20 CTA issue handling process

9.7 Mapping between external and internal OTC TRUE DB

Report accepted, *studied* and *known* are mapped from internal project database to OTC external DB as *investigation ongoing*. The idea of this new status is to make mapping simpler, since customers are interested only about Nokia solving their issues. The most important thing was to indicate that their issues are accepted in Nokia error correction process and another, when issues are solved.

Another new status, *action needed* is not mapped to external OTC DB, since it is wanted to remain in the project server. Benefit of the status appears when errors are mapped from R&D DB into one location under *action needed* status. This is helping issue handlers to find the error reports which need action, more straightforward. Before, errors were mapped under several OTC TRUE statuses. Since a*ction needed* is only temporary status, errors have to move under other project OTC TRUE status as fast as possible.

In addition of mapping, there is a need of co-operation with the developers, who were giving information about the error corrections situation in R&D databases. So, not only the status is giving the information, but the developers have to add detailed description about the error. Status is giving suggestive indication of the error correction status, but it has to be filled with the more detailed information given by developers.

9.8 TSW mapping

Relevant difference to the old mapping was that R&D status is now directed to the one internal OTC TRUE DB status *action needed*. Following status categories are indicating relevant error correction situation in R&D DB, which are needed to be informed forward to the customer by issue handlers; *more info*, *in progress*, *ignored*, *postponed*, *released*, *duplicate*, *verified* and *closed*. All transferred OTC TRUE *report accepted* reports are automatically mapped in to *detected* in TSW (Figure 21).

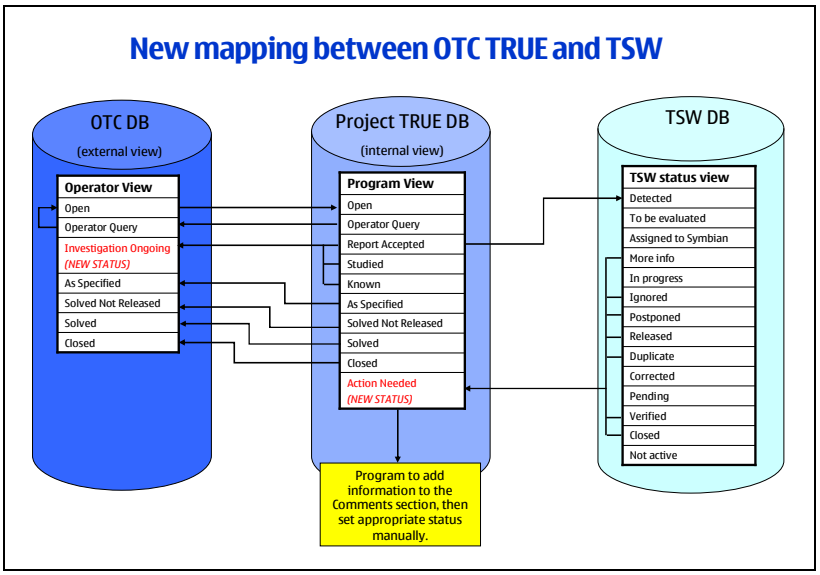

Figure 21 New database mapping between OTC TRUE and TSW

9.9 PCP mapping

Mapping is implemented in PCP similarly as in TSW. All status categories are mapped to one status *action needed* in OTC TRUE DB. Status categories used for mapping in PCP are *released*, *verified*, *closed*, *ignored*, *postponed* and *duplicate*. All transferred OTC TRUE *report accepted* reports are automatically mapped in to *detected* in PCP (Figure 22).

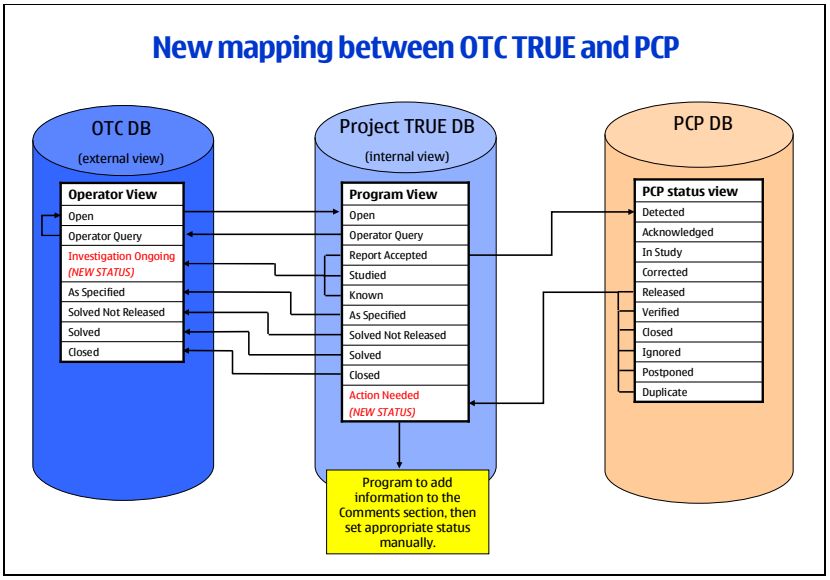

Figure 22 New database mapping between OTC TRUE and PCP

10 Conclusion

The improvement proposal of new status categories and status mapping was introduced for the OTC organization, which was approved by Nokia stakeholders. Improvements were implemented in the autumn 2006. New CTA issue handling process was already taken in practice in the same time as CTA organization was established.

Status categories in the external OTC TRUE view was found to work as expected. However, removing the status *not an error* from internal view was found unpractical, since there were still a lot issue reports which were belonging under this category. Customers were still inserting reports which could not be seen as appropriate error reports. As a result, it has to be considered to return status *not an error* back to OTC TRUE system.

Another big issue was found with the use of *action needed* category and mapping functionality between internal OTC TRUE and R&D databases. After the implementation, there started to be too much error reports under the *action needed* category. This issue was also related to the problem that mapping functionality was not fully implemented. Mapping could not still recognize the actual error correction status of *program integration* team. Mapping was now changing the status according to what was the *S60 common base status* which was not the correct status to follow and which was not giving the right customer error correction indication. This functionality has to be still implemented to the OTC TRUE system in order to issue handling more efficient. Also, reducing amount of R&D status categories from where the mapping is currently initiated needs to be considered.

New location for *region* and *country* issues were added into the OTC TRUE database successfully as well as a new import/export tool was integrated to the OTC TRUE. Tool enabled practically import and export data between excel sheet and OTC TRUE system. On the other hand, CTA statistics and problems related to the Lotus Notes DB use did not improve. Statistics did not still provide accurate information about CTA issues and OTC TRUE had still problems occasionally handle data losses and replication issues.

Problems related to the CTA general work are now identified and some actions are already done. Resources for CTA work has been increased as well as plans for training personnel. CTA is taking stronger role during customer acceptance testing, so that customer issues will be handled better. There is a new guidance document for acceptance reporting for customers and CTA issue handling documentation will ensure that issue management is done according the process. Development work will still continue to solve the remaining problems with the help of the system users and OTC.

The process for CTA issue handling works currently as assumed. It is reproducing and adapting other Nokia error management processes. On the other hand a new development work for an issue and error management system has already started which also include another round for process optimization.

References

Bays, Michael E. 1999. Software Release Methodology. Prentice Hall, Inc.

Customer and Market Operations – Nokia Oyj. [www-sivu] [viitattu 8.3.2007] Saatavissa: http://www.nokia.com/link?cid=EDITORIAL_3828

Ebelin, Charles E. 1997. Reliability and Maintainability Engineering. McGraw-Hill Publishing Company

Enterprise Solutions – Nokia Oyj. [www-sivu] [viitattu 8.3.2007] Saatavissa: http://www.nokia.com/link?cid=EDITORIAL_3833

Haberman, Scott. 1999. Mastering Lotus Notes R5. SYBEX inc.

Haikala, Ilkka & Märijärvi, Jukka. 2004. Ohjelmistotuotanto. Satku – Kauppakaari Oyj.

McDonald, Steve. 2002. Ohjelmistotuotannon hallinta. IT Press Oy.

Mobile Phones – Nokia Oyj. [www-sivu] [viitattu 8.3.2007] Saatavissa: http://www.nokia.com/link?cid=EDITORIAL_3831

Multimedia – Nokia Oyj. [www-sivu] [viitattu 8.3.2007] Saatavissa: http://www.nokia.com/link?cid=EDITORIAL_3827

Networks – Nokia Oyj. [www-sivu] [viitattu 8.3.2007] Saatavissa: http://www.nokia.com/link?cid=EDITORIAL_3837

Nokia Business Group Structure – Nokia Oyj. [www-sivu] [viitattu 8.3.2007] Saatavissa: http://www.nokia.com/A4126325

Patton, Ron. 2001. Software Testing. SAMS Publishing.

Pressman, Roger S. 2000. Software Engineering – a Practitioner's Approach. McGraw-Hill Publishing Company

Pyzdek, Thomas. 2003. Quality Engineering Handbook. Marcel Dekker, Inc.

Platforms and operating systems – Nokia Forum. [www-sivu] [viitattu 28.6.2006] Saatavissa: http://www.forum.nokia.com/main/platforms/index.html

Technology Platforms – Nokia Oyj. [www-sivu] [viitattu 8.3.2007] Saatavissa: http://www.nokia.com/link?cid=EDITORIAL_3845

Tamres, Louise. 2002. Introducing Software Testing. Pearson Education Limited 2002.