HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Electrical and Communications Engineering

Maija Vanhatalo

# Multivariate Modeling in Improving Product Creation Processes

Thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science in Technology

Espoo, 31 January, 2008

Supervisor — Raimo P. Hämäläinen
Professor, Systems Analysis Laboratory

Instructor — Terhi Siimes
Ph.D. (Tech), Nokia

HELSINKI UNIVERSITY OF TECHNOLOGY

**Abstract of the Master's Thesis**


| | | | |
|---|---|---|---|
| **Author:** | Maija Vanhatalo | | |
| **Name of the Thesis:** | Multivariate Modeling in Improving Product Creation Processes | | |
| **Date:** | 31.01.2008 | **Number of pages:** | 7+66 |
| **Department:** | Department of Electrical and Communications Engineering | | |
| **Professorship:** | Mat-2 Applied Mathematics | | |
| **Supervisor:** | Prof. Raimo P. Hämälainen | | |
| **Instructor:** | Ph.D. (Tech.) Terhi Siimes | | |



The modern competitive business environment forces companies to produce new products as efficiently as possible. One of the key areas where productive thinking has lately been applied to, is product development processes. The company will benefit greatly if it can produce more output with same amount of resources as before. The more efficient process is usually achieved when the bottlenecks of the product development process are removed.

This thesis studies Nokia's product creation process. Nokia is the global market leader in mobile device manufacturing creating about 40 completely new models every year. In this study, 64 products launched during the years 2006 and 2007, were modeled using PCA, PLS and batch statistical process control. The target was to create a model that would find the bottlenecks of the process and facilitate decision-making in the product development area while monitoring the process and even predicting the course of it.

The results suggest that PCA is a good method to visualize the general characteristics of a product creation process. On the other hand, due to the unsuitable data, PLS models were not able to produce reliable models of the product cycle time, not to mention the ability to predict the process duration or even any parts of it. Batch statistical process control functioned relatively well in defining the product process path and it was also successfully used for detecting some process bottlenecks. The results of the study were discussed with people working within product development and their comments validated a lot of the modeling results.



**Keywords:** PCA, PLS, batch statistical processes control, R&D process models

TEKNILLINEN KORKEAKOULU

**Diplomityön tiivistelmä**

| | | | |
|---|---|---|---|
| **Tekijä:** | Maija Vanhatalo | | |
| **Työn nimi:** | Tuotekehitysprosessin mallinnus monimuuttujamenetelmillä | | |
| **Päivämäärä:** | 31.01.2008 | **Sivumäärä:** | 7+66 |
| **Osasto:** | Sähkö- ja tietoliikennetekniikan osasto | | |
| **Professuuri:** | Mat-2 Sovellettu matematiikka | | |
| **Työn valvoja:** | Prof. Raimo P. Hämäläinen | | |
| **Työn ohjaaja:** | Tekn.tri. Terhi Siimes | | |


Nykypäivän kilpailuhenkinen liiketoimintaympäristö ajaa yritykset tuottamaan uusia tuotteita niin tehokkaasti kuin vain on mahdollista. Yksi pääalue, mihin tehokkuusajattelua on sovellettu, on tuotekehitysprosessi. Yritys hyötyy huomattavasti, jos se pystyy nostamaan tuotantomääriään, mutta silti käyttämään saman verran resursseja kuin aikaisemmin. Tehokkaampi tuottavuus saavutetaan usein, kun tuotekehitysprosessista saadaan poistettua sitä hidastavat pullonkaulat.

Tämän diplomityön tutkimuskohteena oli Nokian tuotekehitysprosessi. Nokia on maailman johtava matkapuhelinvalmistaja luoden noin 40 uutta matkapuhelinmallia joka vuosi. Tutkimus on rajattu 64:n vuosina 2006 ja 2007 markkinoille saapuneen mobiililaitteen tuotekehitysprosessien mallintamiseen. Tavoitteena oli luoda malli, joka löytäisi tuotekehitysprosessin pullonkaulat ja joka voisi toimia päätöksenteon tukena tuotekehitystyössä toimiville henkilöille. Malli pystyisi sekä seuraamaan että ennustamaan tuotekehityksen kulkua. Mallintamisen työkaluina olivat PCA, PLS ja panosprosessimallinnus.

Tulokset osoittavat, että PCA-mallit toimivat hyvin tuotekehityksen ominaispiirteiden visualisoinnissa. Johtuen puutteellisesta datasta, PLS-mallit eivät puolestaan pystyneet tuottamaan tarpeeksi luotettavia malleja tuotekehitykseen kuluneesta ajasta puhumattakaan siitä, että mallit olisivat kyenneet ennustamaan prosessin tai edes sen osien kestoa. Panosprosessimallit sopivat hyvin tuoteprosessin kulun määrittelemiseen ja prosessin pullonkaulojen havainnoimiseen. Tutkimuksen tulokset esiteltiin tuotekehityksen kanssa läheisessä tekemisessä oleville henkilöille, ja heidän kommenttinsa validoivat osaltaan työssä tuotettuja malleja.



**Avainsanat:** PCA, PLS, panosprosessimallinnus, tuotekehitysprosessimallit

# ABBREVIATIONS

| | |
|---|---|
| BSPC | Batch Statistical Process Control |
| CE | Concurrent Engineering |
| CMM | Capability Maturity Model |
| ES | Enterprise Solutions |
| GPS | Gobal Positioning System |
| M | Milestone/Multimedia |
| MP | Mobile Phones |
| PC | Principal Component |
| PCA | Principal Component Analysis |
| PLS | Partial Least Squares |
| $Q^2$ | The fraction of the total variation of the Y's that can be predicted by a component, according to by cross-validation. |
| $R^2$ | The fraction of the Sum of Squares explained by the model |
| $R^2Y$ | Sum of Squares of all the Y's explained by the extracted components |
| R&D | Research and Development |
| SPC | Statistical Process Control |
| XP | Extreme Programming |

# PREFACE

First of all, I would like to thank my instructor Terhi Siimes for all the support and guidance that she has given me throughout this project. She has truly been a great co-worker and a friend. I wish to thank also Lasse Pesonen, Liisa Saksala and Petri Takala with whom I was privileged to work during this project. I am also grateful for the team leader Mia Talvasto, for giving me an excellent opportunity to work in this project and for encouraging me throughout the process.

I wish also to show my gratitude to my other team members Alina Laasanen, Tommi Lainema, Anna Tuominen, Jarmo Uusitalo, Riikka Veijalainen, Jani Virtanen and Petri Voltti for creating a nice working atmosphere. I am also deeply grateful to the summer trainee of the team, Matti Uronen, for all his comments whether they were related to coding or the contents of this thesis. I would also like to thank my co-worker Simo Salminen for all the encouragement and ideas that he has given to me. I also owe thanks to Jukka Valtanen.

In addition, I am thankful for all the interviewees Matti Alen, Mikko Halttunen, Markku Hiltunen, Eero Juustila, Pekka Laaksonen, Pekka Lähteinen and Aarni Soininen for their valuable comments and ideas concerning the modeling results and the thesis.

I thank also my supervisor Raimo P. Hämäläinen for his guidance.

Finally, I would like to warmly thank my parents, all my friends and especially my brother, Jussi, for all the support and care given during my studies.

Espoo, January 2008

Maija Vanhatalo

# 1. INTRODUCTION

In the highly competitive global business world, the winners are the ones who can differentiate themselves from their competitors and find an ultimate competitive advantage. This advantage can come from any field of product creation. It can be logistics, marketing, user-friendly products or management excellence. What is significant is that in some industries one advantage is not simply enough: To succeed, one has to excel in several fields. This all has led to the point where competitive attitude has reached research and development processes, an area that has been long considered to be without any constraints since that would limit creativity and innovativity which are crucial for new ideas and products.

This competitive attitude has lead to the thinking that everything should be handled as efficiently as possible. Thus, the productivity of each part of the business has been brought out in order to create more profitable products. The productivity of a company will increase when the company is able to produce more with the same resources as measured by output, market position or financial situation. In brief, the company wants to either achieve more with the same input or get the same results with fewer resources. A common way to achieve better productivity is to remove the bottlenecks from the processes that are slowing down the output rate. This is however a somewhat challenging task since there is always something to improve especially in the industries that are under constant growth and change.

This thesis focuses on using statistical multivariate methods to model the mobile phone creation process in different angles in Nokia. These methods cover principal component analysis (PCA), partial least squares analysis (PLS) and batch statistical process control (BSPC) modeling. The aim is to spot the bottlenecks that have been causing problems to different products. The idea is also to create a model that could be used while the products are still in development rather than a retrospectively usable model. In the mobile phone business where customers' needs are constantly changing and the business evolves itself all the time, an efficient product process is clearly a key competence. Since Nokia manufactures around 350 million mobile phones every year, creating about 40 new mobile device models at the same time, the achievable savings in the product process area are significant.

The theoretical part of this thesis will handle some general information about the field of research and development. It covers for example the essential dimensions of

R&D metrics but also the main problems when measuring R&D's productivity. It will also handle the different process models that are used to develop new products all over the world but also in Nokia. In addition, the key variables when developing new products are briefly introduced to explain why these variables were chosen to be used in this study. The selected variables include information about the product development time, the requirements of the product and also the errors that were detected during the product development process.

Besides, the literature part will briefly go through the idea of product platforms, a concept that has become a key component in the creation of new products whether they are software or hardware platforms. The main advantage of a product platform is that even though the development of the platform itself is relatively expensive and time-consuming when compared to standard product development, later on it will provide savings both in time and money. This is because the products that are created on top of proper platform arrive on market faster since their development time is relatively shorter. The creation of a platform however has several key issues and problems that need to be taken in account. These are explained in detail in the literature part of this thesis.

The goal of the practical part of this thesis is to first study the given data with PCA to explain the variation in the data and to see whether there are any factors that group the data or are otherwise significant in some parts of the process. Secondly, a PLS model is used to explain and predict the development time between each milestone. After this, the product process is modeled using batch process modeling to see whether some common bottlenecks can be found inside the processes. It is also interesting to see if these variables can describe the process well enough and to see whether problematic or otherwise deviating products can be spotted from the created model.

If the bottlenecks can be detected early in the product development process and corrected before it is late, savings can be made. The correction of errors and changes in the already done requirements becomes more and more expensive the further the process has advanced. In addition, the model can give a good basis for comparing the creation processes of similar mobile devices, which in turn gives valuable insights for product managers since usually similar projects are carried out in a rather similar way.

A few product managers and other key persons for product processes were also interviewed for this study to find out whether they think that the created models can be useful in practice and to ask if they can suggest any development ideas for the models. Moreover, it was interesting to hear whether they have seen the product development process the same way as the model describes it. Thus the interviews were also used to validate the models.

All in all, the research approach of this study can be considered as *constructive* since the primary target is to develop methods for the problem in question which is now spotting the bottlenecks of the process. This is done in a typical constructive way through an innovative process where the problem is studied with empirical data and verified with interviews. As the secondary target is to create a mathematical model that can be used in the decision making of the company, the research approach can be regarded as *decision methodological*. It is typical too for decision methodological approach that empirical study is used to test the functioning of the model and that the results are considered actually useful in reality[1].

[1] For the definitions of different research methods, see e.g. Olkkonen (1993).

# 2. THE NATURE OF RESEARCH AND DEVELOPMENT

This chapter will discuss the nature of R&D. The concept "R&D" refers here more on product development rather than on the research. Thus, it is more of developing better products from the existing ones than developing something new from a scratch, like a new medicine. Besides to description of the R&D, this chapter will cover the key problems that are faced when trying to measure it. The chapter will end in a short summary of the mobile device industry describing its key characteristics but providing also a brief introduction of Nokia, the current leading mobile device manufacturer.

## 2.1 Key characteristics of R&D

"*In general, R&D activities are conducted by specialized units or centers belonging to companies, universities and state agencies. In the context of commerce, "research and development" normally refers to future-oriented, longer-term activities in science or technology, using similar techniques to scientific research without predetermined outcomes and with broad forecasts of commercial yield.*" [2]

The key reasons why R&D exists in companies are the constantly changing markets and customer needs. Customers' needs change all the time which means that old products will eventually become outdated and thus less desirable. This implies that the company that can respond fastest to the changing trends will collect the pot. In addition, the supply of competitors is always varying, which means that no company can really rest on its laurels but has to keep up the pace. This is where R&D comes into the picture. An efficient R&D makes it possible for a company to be competitive also in the future, when today's products have become just a vague image in history.

Wheelwright and Clark (1992) have summarized well the three driving forces behind R&D. They are intense international competition, fragmented and demanding markets, and diverse and rapidly changing technologies. These are explained in more detail below.

**Intense international competition.** In every business, the amount of global competition has been on the increase. This has made the

[2] http://en.wikipedia.org/wiki/R%26D, accessed 30.10.2007

business more intense, demanding, and rigorous, because companies are punished out of small mistakes immediately.

**Fragmented demanding markets.** The modern consumers have become more sophisticated. When possible, they expect to have personalized products and also the demand for quality is increasing.

**Diverse and changing technologies.** The development of new technologies and better understanding of the existing ones means that there is an increasing amount of solutions available for engineers and marketers in their quest for creating new attractive products.

The three forces mentioned above are common for all industries, but they are especially important for young technologically dynamic industries like the mobile device industry. When the technologies are constantly changing, it is already a challenge to keep up the pace, not to mention when a company should create something new and innovative.

The rigorous global competition, the numerous of market segments and niches and the accelerating technological change have created three major competitive imperatives: speed and responsiveness, productivity, and quality. (Wheelwright and Clark (1992))

**Speed and responsiveness.** The intensive competition and the need to fulfill the customers' changing needs have forced companies to invest in shorter development cycles and better targeted products.

**Productivity.** The exploding product variety and competition in every area has made companies to put emphasize on product development productivity. In new technology companies this amount is larger than in the more mature markets. For example in 2006, an estimate for Ericsson's R&D costs was almost 25% of its revenues[3]. This means that even a small increase in productivity can lead to the saving of millions of dollars.

**Quality.** The intense competition and the demanding customers have put companies in the position where there has to be creativity well combined with overall product quality.

[3] http://www.innovation.gov.uk/rd_scoreboard/, accessed 1.11.2007.

In brief, product development is a major battle area where one has to make big decisions and balance between long term technology decisions and short term product content decisions. This means that the development of a new major technology or platform has to be started well in advance for the products that are to be implemented on top of these technologies later on. A company has to have a long term vision. The success today does not guarantee success later on.

## 2.2 Measuring R&D productivity

Why is the measuring of productivity important? There is a saying concerning the importance of measuring: "*If you can't measure it, you can't manage it*." This means that in order to actually develop a process or even handle it, one has to know the current situation and when a change has happened, one has to be able to measure the outcome.

There are many incentives for studying R&D productivity. Bonsdorff and Andersin (1995, cited on Kerssens-van Drongelen *et al.* (2000)) have summarized them as:

- To *motivate* employees
- To *show employees how they contribut*e to organization's performance
- To *communicate* performance expectations
- To provide *management information*
- To identify *performance gaps*
- To support decision making

These reasons can be roughly divided into two categories: prospective metrics and retrospective metrics (Tipping and Zeffren (1995)). Prospective metrics aim to detect the bottlenecks during the development process and retrospective metrics, for one, try to create improvement ideas based on the historical data for the overall R&D process. When we add to this time dimension the performance aspect, which includes variables like quality, timeliness and cost, in addition to the organizational goals, we achieve the whole taxonomy of R&D metrics. This taxonomy has been collected by Kerssens-van Drongelen *et al.* (2000) from several sources and it is summarized in Figure 1.

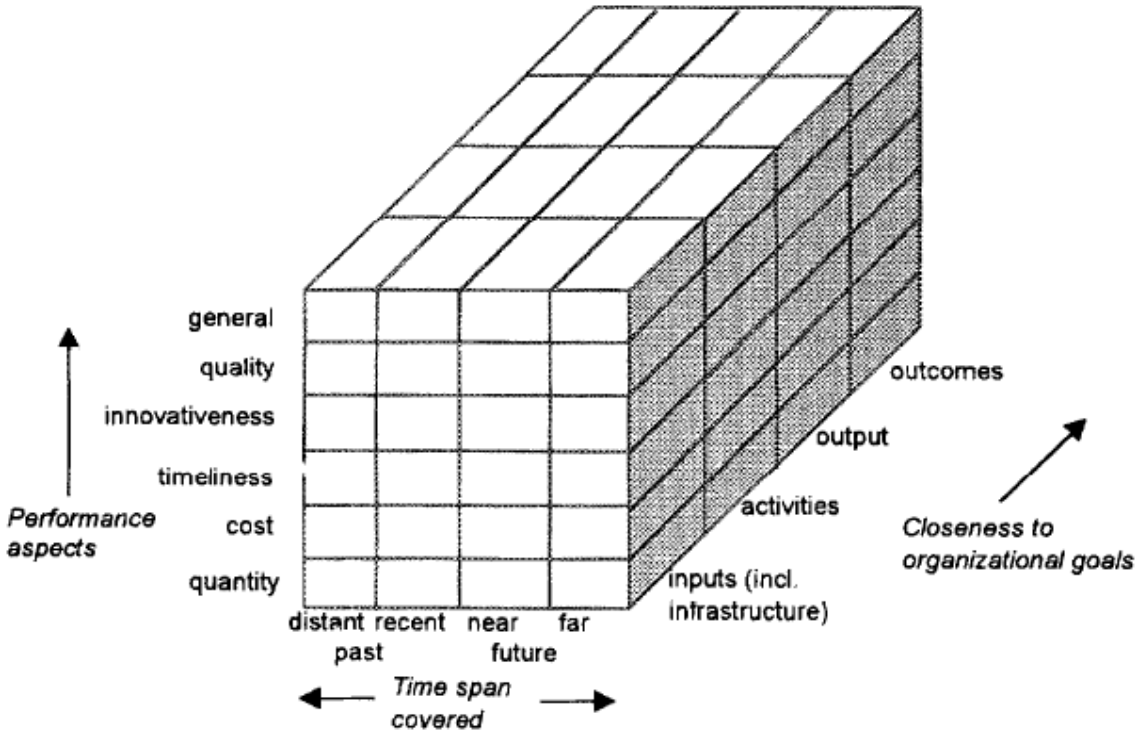


Figure 1: The different dimensions of R&D metrics taxonomy (Kerssens-van Drongelen *et al.* (2000))

## 2.3 Problems when measuring R&D

There are numerous difficulties when measuring R&D mainly due to its special nature. The biggest problem is, without a doubt, time. The time lag between the initial R&D investment and the emergence of results makes it difficult to establish a connection between R&D efforts and their payoffs in the marketplace. This time lag can especially be seen in basic research but it is also quite significant in development projects. Because of the time lag, the corrective actions become difficult to apply since when for example the financial outputs are at hand it is usually too late to change anything within the current project. (Kerssens-van Drongelen *et al.* (2000), Helo *et al.* 2000)

The second problem with the measuring is matching specific R&D inputs like money, man-hours and intermediate outputs, like new technologies, with final outcomes, the products. It is problematic to tell which previous products have influenced the current product development, because knowledge is also gathered from the failed products, which are easily considered worthless. The failed products can for example give valuable information of what R&D should not do and how the process should be changed in order to work better. These kinds of results are rarely

measured even though they can greatly benefit the company. (Chiesa (1996), Kerssens-van Drongelen *et al.* (2000))

The third problem lies within quantification. Not all the benefits that R&D generates are quantifiable. For example the high-tech image that a company reaches due to its R&D may attract new high potential employees, but it would be extremely difficult to put a price tag for this. (Kerssens-van Drongelen *et al.* (2000))

The next problem is the accurate isolation of the contribution to success (Hodge 1963). This means that the success is a sum of several factors. The marketing team for example is crucial for success, but still if the product development has been executed badly, it does not matter how well the marketing department is doing its job.

The fifth problem is the broad variety of different programs (Kerssens-van Drongelen *et al.* (2000)). Firstly, there are four stages in R&D: basic research, applied research, prototype/pilot plant and commercial development. In addition to this, within e.g. commercial development there are numerous different sub-projects because products are aimed at different customer segments. It is natural that it takes less time and effort to develop a low-end product than a high-quality high-end product. This makes all the products quite exceptional. The comparison of a low-end mass-customized mobile phone and a high-end multimedia device is already challenging even though they are both mobile phones and created by the same company not to mention products in different business fields.

The final obstacle for R&D measurements is the acceptance of performance measurements. In general, the process is considered to be innovative and measuring this delicate process could harm the innovative spirit and lead to counterproductive results because of decreased motivation etc. (Brown and Svenson (1998))

In addition to all problems listed, it is also noteworthy to mention that even though a lot of research has been carried out in this field, a systematic and empirical research into the consistency among R&D measurement approaches and contingencies has still not been made (Kerssens-van Drongelen *et al.* (2000)). Even if there are considerable problems with measuring R&D, it still needs to be done and it has already become a competitive advantage for certain companies. Like Francis (1992) has put it:”The time has come to lay aside the old excuses for not measuring and baselining R&D effectiveness, and do it anyway”.

## 2.4 R&D in the mobile phone industry

The mobile phone industry like majority of the technical industries, is a highly **competitive business** and not for nothing. According to Gartner's studies[4], it is estimated that 1.13 billion mobile phones are sold during 2007. What is also noteworthy, is that today's winner is not necessarily on top tomorrow. This can be easily seen from history. Even though Nokia has led the industry since 1998, the market shares have been changing drastically between the different competitors after this. One example is Motorola's Razr. After a highly successful period Motorola's sales are today going down[5]. As it has been explained in many articles: "One-hit wonders don't work in today's global wireless market, where mobile phone makers need iconic devices that appeal to different cultures and sell at a variety of prices."[6] Because of the short life-cycle of products, new and personalized phones have to be developed all the time.

Thus, it is evident that the market is highly competitive. Competitors adopt fast the same features or designs that others are using and the slower adopters will easily suffer great losses. The clamshell phone is a good example. In 2004, Nokia was not fast enough to follow the clamshell trend and lost market share significantly[7].

The **fast technological changes** and new innovations are also a very typical factor for mobile phone industry. Ten years ago, the major features in a mobile phone were the ability of sending messages and making phone calls. Today, one can watch TV, take high quality pictures, navigate with GPS, surf the Internet and much more not to mention all the network changes that have occurred. That is why mobile phones are nowadays called multimedia computers not just plainly phones. A good example of the impact of new technologies and changing needs is that Nokia, a mobile phone company, is currently the biggest digital camera manufacturer in the world.

Mobile phone companies have two totally different **customer segments**: operators who sell phones to customers and then the actual mobile phone users. The role of

---

[4] http://www.digitoday.fi/mobiili/2007/08/23/Motorola+purki+varastoja+alennushinnoilla/20072038466, accessed 30.10. 2007.
[5] http://compoundsemiconductor.net/cws/article/news/27633, accessed 30.10. 2007.
[6] http://money.cnn.com/2007/07/27/magazines/business2/motorola_analysis.biz2/index.htm, accessed 30.10. 2007.

[7] http://www.hindu.com/biz/2004/04/19/stories/2004041900451800.htm, accessed 30.10. 2007.

operators varies a lot depending on the country. In some countries, the operator is better known than the company that is manufacturing the devices. The role of operators is thus crucial to mobile phone companies. However, the symbiotic life of mobile phone companies and operators has been endangered lately. Nokia for example, is entering the operators' core business with the "Ovi" service that will provide music, navigation and game services.

It must be emphasized that mobile phones target a mass-market of consumer, enterprise and professional users. People ranging from young to old are using mobile phones in the extent that in some countries the penetration rate of mobile phones is even 140 % [8].

### 2.4.1 Nokia

The target company of this thesis, Nokia, is the largest mobile device manufacturer with about 39% market share (Figure 2). The total worldwide mobile device sales in 2007 is estimated to be 1.1 billion pieces out of which Nokia will sell over 350 million pieces. The net sales of Nokia in 2006 increased to over 41 billion euros and the operating profit was over 5.4 billion euros. Currently, the R&D costs represent 9.5% of the net sales. (Nokia 2006)

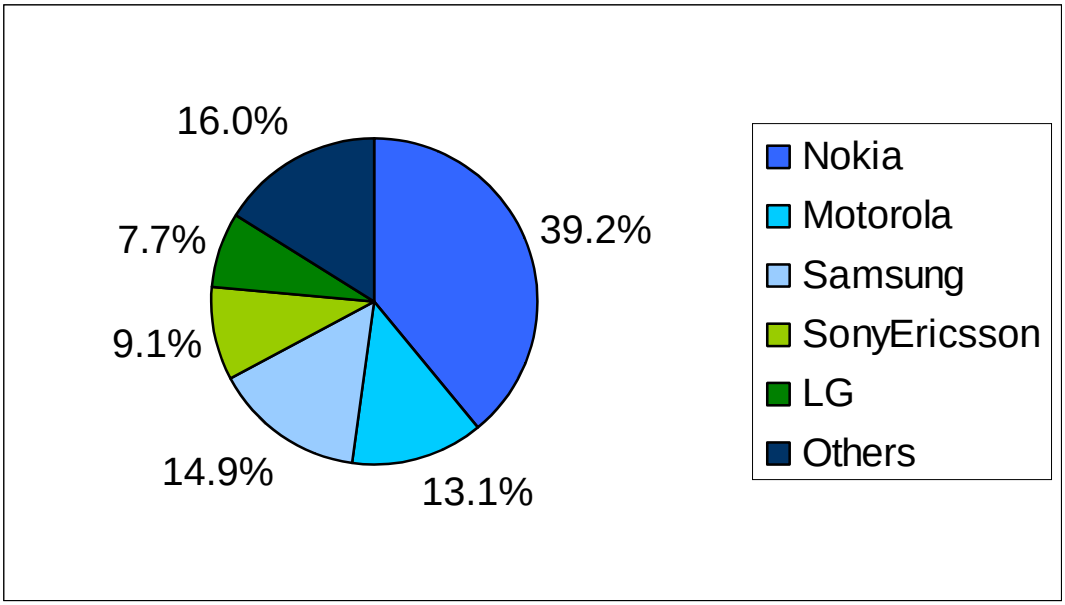


Figure 2: The market shares of the biggest mobile phone companies[9]

[8] http://communities-dominate.blogs.com/brands/2007/01/putting_27_bill.html, accessed 30.10. 2007.
[9] http://www.mobilemonday.net/news/shares, accessed 30.10. 2007.

The R&D produces over 40 new models every year[10] that come from three different business groups: Mobile Phones, Enterprise Solutions and Multimedia. They are defined in the following way [11]:

"**Mobile phones** connects people by providing expanding mobile voice and data capabilities across a wide range of mobile devices."

"**Multimedia** gives people the ability to create, access, experience and share multimedia in the form of advanced mobile multimedia computers and applications with connectivity over multiple technology standards."

"**Enterprise Solutions** offers businesses and institutions a brad range of products and solutions, including enterprisegrade mobile devices, underlying security infrastructure, software and services."

As a summary, Mobile Phones creates phones for the mass markets, Multimedia for high tech people and Enterprise Solutions for business environments.

[10] http://www.nokia.com/A4136001?newsid=1099952, accessed 30.10.2007.
[11] http://www.nokia.com/A4126325, accessed 30.10.2007.

# 3. PRODUCT PLATFORMS

The concept of product platforms is maybe the biggest renovation that has occurred in the last few decades in R&D. Its impact on product development has been great and the idea of platforms keeps on gaining constantly more ground. This chapter will briefly go through the definition of platforms and the pros and cons of using them. It will end by describing the current situation in the mobile phone industry.

## 3.1 Defining the product platform

The concept of product platform has been defined in several ways in the literature. According to Meyer and Lehnerd (1997) it is “a set of subsystems and interfaces that form a common structure from which a stream of derivative products can be efficiently developed and produced”.

Simpson *et al.* (2001), on the other hand, have a wider view. The platform is not anymore just a set of mechanical components. They consider a platform to be “a set of parameters (common parameters), features, and/or components that remain constant from product to product, within a given product family”.

Robertson and Ulrich (1998) go even further. They state that a product platform is “the collection of assets that are shared by a set of products”. These assets can be components, processes, knowledge and even people and relationships.

According to Muffato and Rovenda’s study (2000), there could be yet another branch of defining a product platform. They suggest that a product platform is a multifaceted concept affecting:

- Production and logistics processes (costs, investments, operations complexity, etc.);
- Development process (development lead time, standardization, quality and reliability of design);
- Project organizational structure (teamwork, design task partitioning, relationships with suppliers);
- Knowledge (know-how transfer between projects, influence on and by technology, etc.).

As can be seen above, the ‘product platform’ is ambiguous term and it has not found its definite place yet. Depending on the writer, it can either cover only the common physical parts or include the whole process in it. However, in this thesis, a narrower definition is used: *The product platform is the common technology basis of all individual products within a product family*.

Modularization is a term that often comes up with platforms and the two concepts actually partly overlap each other. They both are vaguely defined and even in the academic world, the definitions vary significantly. Usually the modularization refers to the use of independent units, modules, to create product variants. The modularization helps also to achieve the same benefits e.g. reduced cost and faster development time for upgrades. Generally, the biggest difference is maybe that the designers of product platforms usually think that their platforms utilize modularization, but people working with modularization rarely mention platforms.

## 3.2 The benefits of using product platforms

One of the biggest benefits brought by product platforms is a shorter time to market. The upgrades are faster to build as well as are any enhancements. By reusing the existing assets that already have been tested and verified, the product development avoids doing the same work load again and the products are more reliable (Robertson and Ulrich (1998)). Sony Walkman is maybe the most familiar example of product platform usage (Sanderson and Uzumeri (1995)). With platform thinking, Sony has been able to manufacture products as fast as its competitors, but what is more, the lifecycle of Walkmans has been higher due to the high quality platform.

Thanks to platforms, customers will also get more differentiated, personalized products, since it is easy to make few changes on top of a well-planned platform. The mass customization results a high volume manufacture of individually tailored products which has proven to be a key competitive advantage in the new product development world. A good example of this is Compaq, which was able to produce their computers for both corporations and homes, based on one basic platform resulting in that only 2 % of their sales were invested in R&D (Meyer and Lehnerd (1997)).

Cost reduction is also a significant driver for platforms, since development costs are lower thanks to high volumes and small diversity of components. Thus the company

will achieve benefits brought by economies of scale. Muffato (1999) suggests that in the manufacturing environment, cost reduction in capital investments can rise as high as 50 %. In addition, product platforms enable improved organizational learning and knowledge management, thanks to the common platform. This reduces risks too, because employees learn from past experiences and re-work is diminished. What is more, the shift from one product to another is easier since the core of the platform is familiar for everyone. It is also acknowledged that the platform approach reduces cannibalization to a certain degree by separating the products further (Krishnan and Gupta (2001)).

All in all, higher productivity is achieved with platforms since the amount of work is reduced, but also the quality is better, costs are lower and product development lead time is reached. Platforms also enable mass customization and thus better customer understanding.

## 3.3 The disadvantages of using product platforms

Product platforms are highly praised in the literature. However, their disadvantages are somewhat less explored (Halman (2003), Krisnan and Gupta (2001)). The main challenge with product platforms lies in their development process. As Robrtson and Ulrich (1998) as well as Meyer and Lehnerd (1997) have pointed out, it is difficult to balance between commonality and differentiation (Figure 3). The customers demand customized products, but the more you customize the more expensive the platform and, in the end, the product will become. When the product family includes both high-end and low-end products, a problem arises. If the platform is under designed, the quality of high end products suffers. On the other hand, if the platform is over designed the economies of scale achieved through platform thinking, will disappear since the over-design hinders profitability (Krishnan *et al.* (2001)). This means that a badly designed platform will destroy the profitability of the whole product family. Thus, the development is crucial. Hauser (2001) has even concluded that the utilization of platforms might have negative correlation on profitability because of the problems mentioned above.

Still, when companies try to avoid having too diverse products under the same platform and the product family by creating several platforms, the next problem emerges. The platform development work is beneficial only if there are enough

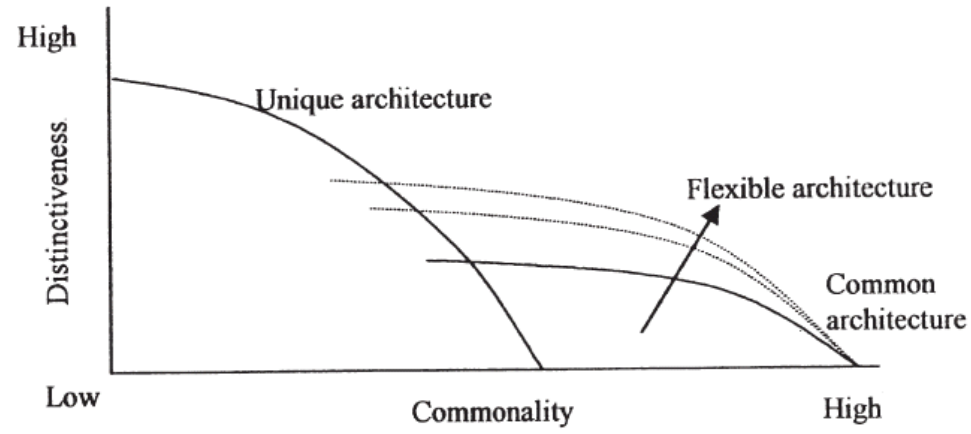


Figure 3: Commonality vs. distinctiveness in modularity (Seriff (1998), cited in Muffato *et al.* (2000)).

products using it. Lately, the target has clearly been to reduce the amount of platforms since one platform per product is not profitable, because platform development takes more time and resources than just producing individual products (Halman 2003). Naturally, this is due to the fact that platform is a long term investment and it has to include future trends and technologies in it in order to be competitive also in the future. The nature of platforms is also responsible for the delayed return on investment, since the development time is longer. This is why platforms are not the most suitable solution for all markets and firms (Halman 2003). What is more, the company cannot rely on one platform for long. The development of platforms must be continuous all the time. (Meyer and Lehnerd (1997), Meyer (1997)) Especially, when a company is competing in an innovative environment, the lifecycle of a platform can not be long. However, as Uzumeri and Sanderson (1995) have pointed out, a company is forced to decide between the model variety and the rate of change, since the resources to finance these strategic decisions come from the same pool (Figure 4).

There are also organizational problems with the design of platforms. On the other end of the process, there are marketing managers and product planners who emphasize customers and their needs in different market segments. On the other end, there are system-level designers addressing the problems with the product architecture and what components and modules should be used in the designing. This means that the process needs a lot coordination between the company’s marketing, design and manufacturing functions. These functional groups may not be accustomed to working with each other, so conflicts can arise over differing time frames, goals and assumptions. (Robertson and Ulrich (1998))

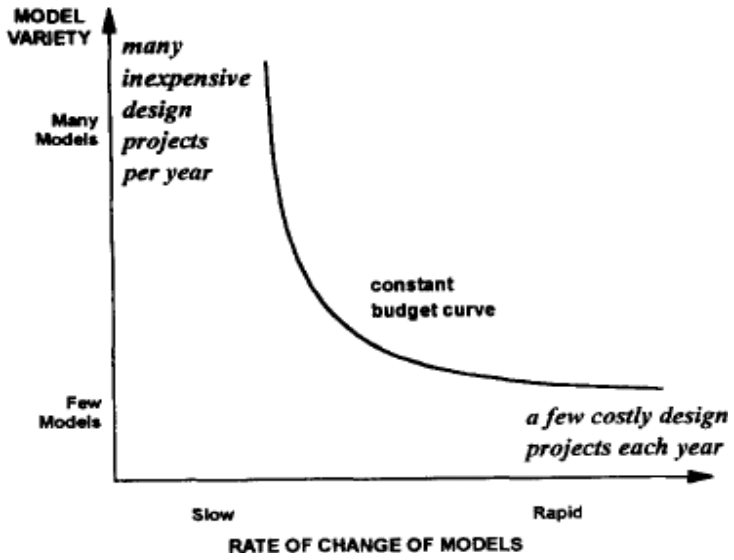


Figure 4: Model variety and the rate of change of models. (Uzumeri and Sanderson (1995))

## 3.4 Platforms in the mobile phone industry

The challenges that lie in the mobile phone business and platform development derive from the customers. The mobile phone industry is perhaps one of the fastest evolving and growing businesses. Before the early 1990's, mobile phones were large and they had to be carried in jacket pockets, so they were mainly used as car phones. Nowadays the devices include all kinds of features ranging from GPS to digital video cameras and Internet access. Since the technologies and customer needs are constantly changing, it is highly demanding to create a platform that would be up-to-date even after a few years. As mentioned above, the platform development starts very early and it should cover a wide range of products. Thus, predictions should be rather accurate or otherwise the whole product family might suffer.

The mechanical part of a platform poses problems since marketing managers have clear definitions how small and light mobile phones the customers want. This limits the mechanical design significantly and puts a lot of constrains on the component size and interfaces inside the phone. This makes the design inside a product family challenging, not to mention the products that are going to be released in the markets in the future. (Vilkman 2002)

The platform thinking has been mainly utilized in traditional engineering e.g. in printers, cars, and Walkmans. In these products the technology has had some specific standards. The tape in Walkmans and paper size in printers are fixed-sized. This

means that the technology and the customer needs are also rather constant making platform development easier than in mobile phone business.

In addition, mobile phones have *two* different platforms included in them. There is the mechanical platform, hardware (HW) platform, including all the components like the engine, the display and the battery, but there is also the software (SW) platform on top of that. The literature tells rather much about both of them separately but rarely merges them together.

### 3.4.1 Software platforms in mobile devices

There are altogether five different software platforms that Nokia is currently using: Series 30, Series 40, Series 60, Series 80 and Series 90. In these days Series 40 and Series 60 have risen to be the most important ones, since Series 30 is used in the very basic mobile phones, Series 80 is only in communicator devices and Series 90 in a very few specific models.

Series 40 (S40) is the most utilized software platform with hundreds of millions users. However, S40 is only used at Nokia, whereas Series 60 (S60) is also used by Samsung, LG Electronics, Panasonic, Lenovo and Siemens. S40 has been launched some years before S60 and nowadays it is mainly utilized in mass-market phones since it does not support as many applications that S60 does. Technically this means that S40 developers have been creating applications mainly with Java™ and Flash Lite from Adobe but with S60 this can be done with C++ and Python as well. S60 is thus more suitable for smartphones that are used in business environment and are loaded with diverse applications from music to games.[12] [13]

[12] http://www.forum.nokia.com/main/platforms/s40/index.html, accessed 30.10. 2007.
[13] http://www.forum.nokia.com/main/platforms/s60/, accessed 30.10.2007.

# 4. R&D PROCESS MODELS

Many of the process development models have originally been designed for software development. However, they can easily be applied to all kinds of product development processes. The purpose of these process models is to provide understanding about the system that is being developed and at the same time help managers to coordinate the development activities.

This chapter will describe a few of the most well-known process models and in the end depict shortly the process model used at Nokia. The target of this chapter is to give a general idea of the process models and thus reveal that in different part of the process different variables are important since each milestone has its own role in the bigger picture.

## 4.1 The waterfall model

The waterfall model, originally created by Royce (1970), is probably the earliest product development model. This model consists of a linear series of phases, starting from the definition of requirements, going through the phases of design, implementation, verification and ending with maintenance (Figure 5). According to the literature source, the amount of phases and their naming can vary but the main idea is always the same.

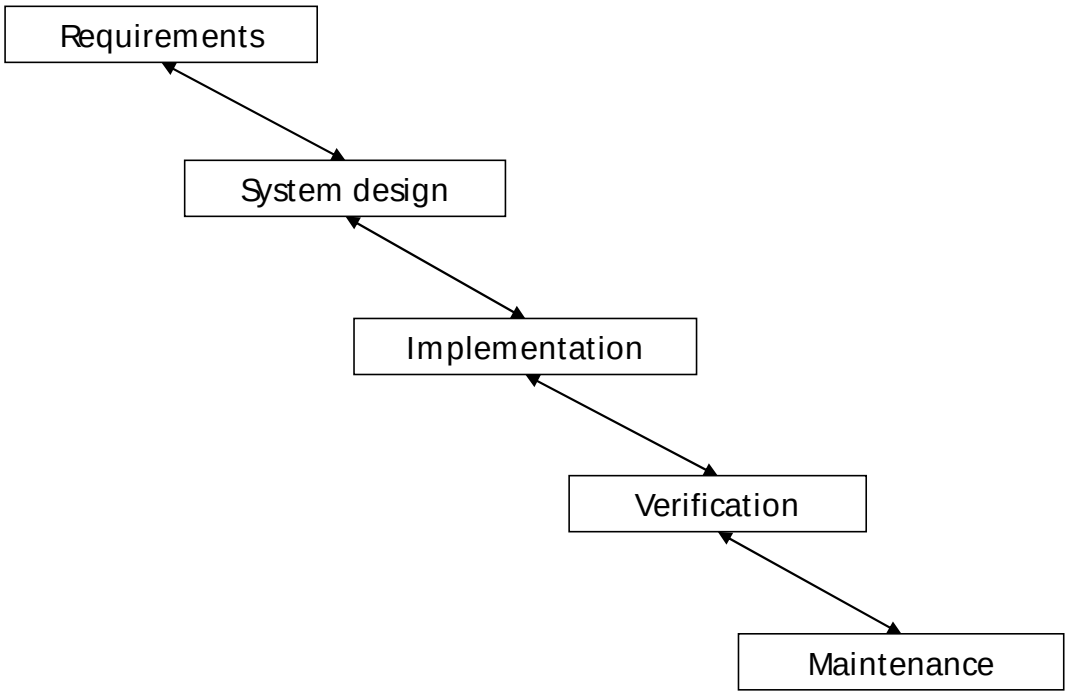


Figure 5: An illustration of the waterfall model

The key benefits of this model are a clear structure, the ease of control and management and the fact that it is widely used. The waterfall model has even formed a base for several development standards. However, there are also some disadvantages. One of them is the lack of flexibility to accommodate specification and design changes during the project (Cusumano and Smith (1995)). Integration and testing, which are carried out at a late stage of the process, may cause unexpected delays to the project launch. Especially in the telecommunications industry, where both hardware technologies and customer requirements change rapidly, the full set of requirements is difficult to define completely (Perttula (2007)).

## 4.2 The iterative development model

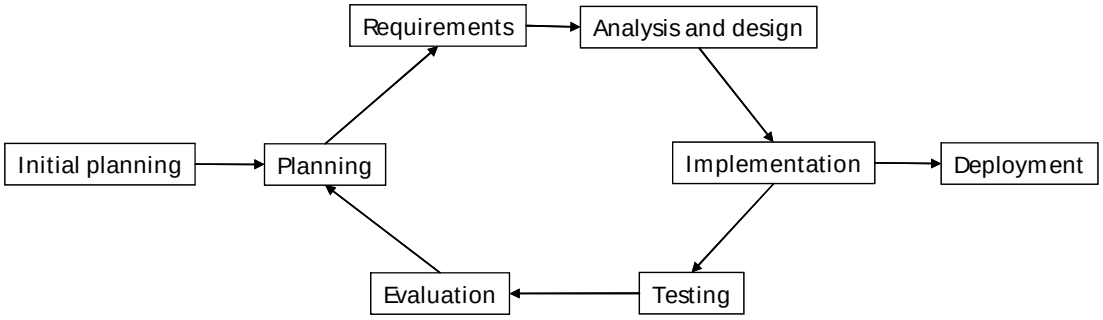


Figure 6: An illustration of the iterative and incremental development process[14].

The iterative development model has been developed to solve some problems that the waterfall model is having. The key benefit of the iterative model is achieving faster development cycles. This is because of quicker results, the usage of less up-front information and the offer of greater flexibility (Perttula (2007)).

In every cycle of the iterative model (Figure 6) common development methods like planning, implementation, testing and verification are executed. Each iteration cycle makes the product readier and also reduces the risk of developing a product that does not meet the product requirements. If new requirements need to be added to the product, a new iteration round will be made.

One of the main advantages is that the project is divided into small iteration projects i.e. rounds instead of a one massive project. The result is that these small projects can be integrated and verified separately before the final integration and verification. In brief, each iteration round is like a small waterfall model with the additional input information from the last cycle helping during the next cycle.

However, in spite of the good aspects, there are also a few challenges to point out. Fujii and Kambayashi (2002) suggest that the model is not efficient because of the changes created by each iteration round can cause a significant amount of rework. This means that the product should be as modularized as possible and the architecture well planned, so that the new changes would not have such a dramatic impact on the product. Albany, (1998), on the other hand states that the communication and coordination effort between the development iteration rounds and with the customers can easily cause delays in the project (cited in Perttula (2007)). Although customer involvement is a good thing in a product development process, it can easily bring more and more improvements to the new product which naturally delays the product launch. What is more, Eppinger (2001) has found out that because employees working on a new project know that new iteration rounds will probably come, they will not put too much emphasis on the first iteration round. All the iteration rounds should be thus planned beforehand and tightly managed in order to minimize this kind of effect.

## 4.3 Agile development models and the extreme programming

Agile development models are mainly used in software development. The main idea of these models is to promote several iteration rounds throughout the life-cycle of a project. At the end of each iteration round, the product release should ideally be ready to market, which means above all that there are no bugs left. The more complex the product is, the more iteration rounds there are to add new features and better the functionality. Because of the constant development of the product, the agile methods are suitable for teams facing unpredictable or rapidly changing requirements. However, this flexibility requires face-to-face communication and a small team around the project, so that no information is lost between the numerous iteration rounds that can only last a few weeks. Since a lot of information is changed verbally, agile methods are often criticized to be undisciplined, because of lack of proper documentation.

The extreme programming model (XP), created by Kent Beck in 1996, might well be the most famous agile development model. Its relation to the “old” waterfall model and the iterative development model is depicted in Figure 7. XP has proven to be very efficient compared to the older models. For example Spayd (2003) discovered

[14] http://www.iscn.at/select_newspaper/procurement/tieto.html, accessed 30.10.2007.

that XP was able to reduce the product cycle time from one year to 90 days. There are however limitations with XP too. Paulk (2001) has found out when using a CMM (Capability Maturity Model) that it does not work well when applied to bigger projects with large teams. Also high reliability might be hard to achieve and the lack of documentation poses some problems as well.

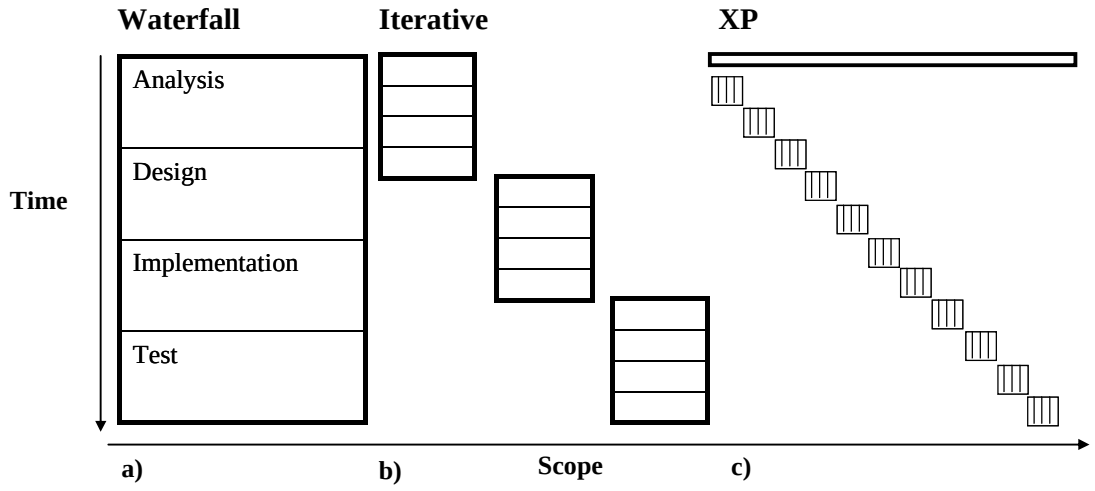


Figure 7: a) The waterfall model b) The iterative model and c) The extreme programming model (adapted from Beck (1996))

## 4.4 The Stage-Gate Process

As Thamhain (1995) has put it, the definition of the stage-gate process is: "A series of per-defined project phases, called *stages*, each leading into a *gate* with pre-defined criteria for overall project success". The idea is illustrated in Figure 8.

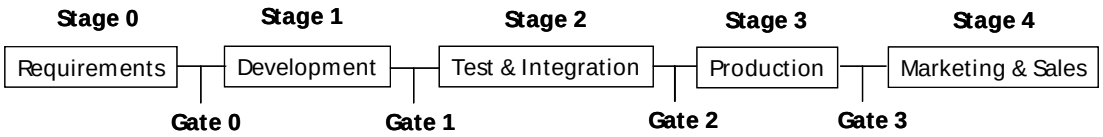


Figure 8: The stage-gate process (adapted from Thamhain (1995))

The stage-gate process is especially used in the development of new products where it serves as a procedural template. Each stage defines the main scope of work, the deliverables and the procedures how the deliverables are reached. The gates on the other hand specify the criteria of overall project success where the emphasis is on the

latest stage. This means, that when the project is running, the employees are concentrating on these gate criteria.

## 4.5 Concurrent Engineering

Concurrent Engineering (CE) was originally developed by Winner *et al.* (1988) for the creation of military equipment (cited in Ketola 2002) form where it has spread on to other industries. The main idea behind it is to carry out the execution of project activities in parallel in order to get shorter project cycle time. Other major benefits are higher quality and better productivity. Table 1 summarizes the benefits on different areas from the results reached in the studies of Boeing, IBM, AT&T and HotPoint (Medhat and Rook (1997)).

Table 1: The benefits of the CE model compared to the sequential development model. Results are from companies Boeing, IBM, AT&T and Hotpoint. (Medhat & Rook (1997))

| **Outcome** | **Contribution** |
|---|---|
| Development time | 30% less |
| Engineering changes | 65 % fewer |
| Time to market | 20% less |
| Overall quality | 200% higher |
| Productivity | 20% higher |
| Sales | 5% higher |
| Return on assets | 20% higher |

The product development process here is divided into smaller development tasks like HW, SW, mechanics, marketing and manufacturing. These entities are then executed in subprojects. In order to have a good view of the project as a whole, the parallel activities are usually synchronized together at checkpoints that are most generally called as milestones, phase reviews or gate reviews. At these checkpoints, a review is made whether the targets of the phase were reached or not and whether the project is ready to move on to the next phase. The overview of a CE process compared to the more traditional sequential engineering process (e.g. waterfall model) has been depicted in Figure 9 (Medhat and Rook (1997)).

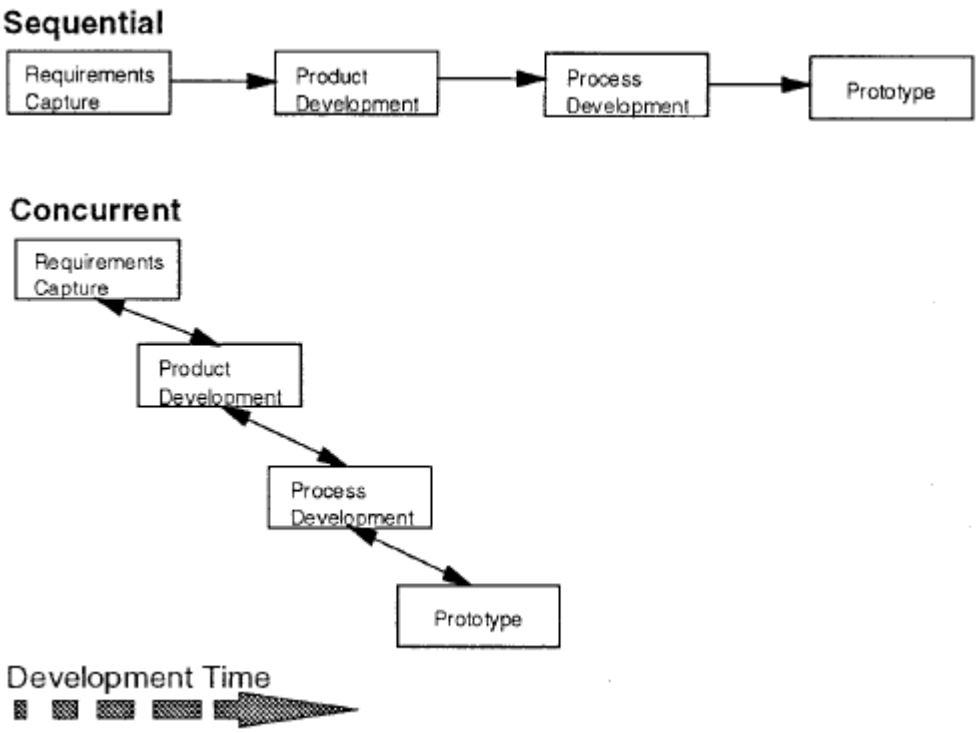


Figure 9: Sequential versus Concurrent Engineering Development cycle. (Medhat & Rook (1997))

A key requirement for the success of a CE process is cross-functional teams since several departments must collaborate over the whole project life-cycle to ensure that the outcome truly reflects customers needs and desires. The benefit of a cross-functional team is that it fosters a work environment of strong interfunctional collaboration and involvement, lowering the traditional organizational barriers, and making the project organization more transparent which enables for one integrated project planning, stage-gate processes and self-directed team organizations. (Thamhain (1995))

There are also some challenges that are associated with the CE process. The biggest of them is the lack of co-operation between different teams. The CE approach does not encourage different teams to have a dialogue with each other before the product integration phase. There might also be problems with resource sharing between the groups and with collective multi-functional decision making (Thamhain (1995)).

## 4.6 The product development process in Nokia

The Concurrent Engineering (CE) process is the foundation of Nokia's product development process (descripted also in Ketola 2002). This is due to the fact that in the mobile phone business short time to market is one of the biggest competitive

advantages and concurrent engineering serves well in this field as mentioned above. The stages/ milestones of the product development process have been listed and explained below in Table 2. The milestones M0-M4 represent the CE development process. The milestones M5-M9 for one represent the product engineering process the purpose of which is mainly to maintain and improve product quality and competitiveness and finally to close the whole development project (Pesonen (2001)).

Table 2: First six milestones

| **M0** | | **M1** | | **M2** | | **M3** | | **M4** | | **M5** |
|---|---|---|---|---|---|---|---|---|---|---|
| | Specify product | | Design | | Implement & integrate | | Test | | Launch | |

**Concurrent Engineering**

M0 Product specification

M1 Design

M2 Implementation and integration

M3 Testing

M4 Launch

**Product Engineering**

M5 Transfer to product engineering

M6 Ramp-down proposal

M7 Volumes frozen

M8 End of deliveries

M9 End of product life cycle

# 5. THE KEY VARIABLES IN R&D PROCESS

The target of this chapter is to go through the key variables of an R&D process. This is to show, why these variables are important in R&D and why a lot of attention is being put on them. This chapter aims also to explain their nature and revel why they are used as the data of the modeling part.

## 5.1 The product requirements

As Kotonya and Sommerville (1998) have summarized it, requirements are "A statement of a system service or constraint". The requirements in general have been created to work as an interface between the customer who wants to have a new product and the executing personnel who will create the product according to the customers' needs and desires. Due to this, requirements are defined in the very beginning of product development as a specification of what should be implemented. Requirements are descriptions of how a system should behave, they can be application domain information, constrains on the system's operation, or specifications of a system property or attribute. Kotonya and Sommerville (1998) add that requirements can be divided into five subgroups as followed:

- Very general requirements which are created to depict what the system, for example a machine or a program, should do.
- Functional requirements which define one of the system's functions.
- Implementation requirements which are meant to describe, how the requirements should be implemented.
- Performance requirements that specify a minimum acceptable performance level for the system, for example how fast a machine should function.
- Usability requirements that set out the maximum acceptable property to demonstrate the use of the system for example how fast the system reacts to the user's commands.

As with every field, also with requirements there are some disagreements of definitions. Some people suggest that requirements should be statements of what a system should do instead of how it should do it. The latter is however often easier for implementing engineers to comprehend than abstract problem definitions. In

addition, sometimes requirements are limited by the execution environment they are facing. In such cases, the implementation constraints must be specified in order to ensure compatibilities with the new environment. What is more, every now and then the specifiers of the new system are experts in the application domain where the system will be used. Because of this, requirements can also be descriptions of how to perform a certain computation using this special application domain data. However, it should be borne in mind that the most important thing with requirements is to include all the necessary information in them, so they can be implemented as the customer has originally intended. (Kotonya and Sommerville (1998))

Since requirements are the key concept to outline a new product, the problems in executing them are rather significant. It has even been estimated that the cost of fixing a requirement error can be up to 100 times the cost of fixing a simple programming error. This is due to the fact that when altering a small detail somewhere in the system, it automatically changes all the other parts that depend on it. Practically, this can even mean that the design, implementation and testing may have to be carried out again. In brief, the earlier a defect can be detected, the less rework its fixing will most likely generate. The four main problems emerging from the use of requirements are listed below (Kotonya and Sommerville (1998)):

- The defined requirements do not actually reflect the real needs of the customers for the system.
- The execution of requirements can be inconsistent and / or incomplete.
- It is expensive to make changes to requirements after they have been agreed since changing one detail in one part can easily affect other parts in the system.
- There can also be misunderstandings between customers, those creating the system requirements and software engineers developing or maintaining the system.

So, why is it so crucial to get the requirements right in the beginning? As mentioned above the change process is expensive and when requirements are executed incorrectly, the budgeted costs arise easily above the target. In addition, customers or end-users might not be satisfied with the system or product. They may not utilize its facilities or they may even decide to cast it aside altogether. What is more, the system might be unreliable in use, with regular crashes disrupting the normal

operation. Also, if the system continues to be in use, the costs of maintaining and updating it are usually high. (Kotonya and Sommerville (1998))

Requirements can be roughly divided into two categories, functional and non-functional requirements. As the name suggests, the non-functional requirements are not specifically concerned with the functionality of a system. These requirements include mainly safety, security, usability, reliability classes and the main reason why they differ from functional requirements is that they place restrictions on the product being developed and the development process itself. They also identify the external constraints that the product must meet. In brief, they tell more of what the ready product should not do while the functional requirements specify what it should do. (Kotonya and Sommerville (1998))

As the non-functional requirements include constraints related to for example quality, resources and timescales of the system being developed, this restrictive nature makes them important. This means that functional requirements are first sacrificed in order to meet the non-functional constraints. However, not all the non-functional requirements hold the product back. Some of them restrict more of the development process e.g. development standards, methods and implementation languages. In addition, some non-functional requirements are not directly associated with the product or development processes but rather arise from external constraints outside the enterprise. Nevertheless, the most significant ones are legal, economic and interoperability constraints.

In short, the non-functional requirements can be classified in three classes: product requirements, process requirements and external requirements (Kotonya and Sommerville (1998)). A little more detailed listing of different requirements categories can be seen below as the ‘IEEE-Std 830-1993’ has specified them. The non-functional requirements are from 2 to 14 in the list.

1. Functional requirements
2. Performance requirements
3. Interface requirements
4. Operational requirements
5. Resource requirements
6. Verification requirements
7. Acceptance requirements

8. Documentation requirements
9. Security requirements
10. Portability requirements
11. Quality requirements
12. Reliablity requirements
13. Maintability requirements
14. Safety requirements

Even though there are clear defined classes for requirements, the reality is not in every case just as simple, and the classification often depends on the way of defining. Sometimes a bit more detailed definition between a system customer and a system developer can change the whole categorization. This is because of the special nature of non-functional requirements. They are not covered as well as functional requirements in the requirements engineering methods. Existing methods are usually based on for example the functional analysis and thus are inherently limited as far as non-functional requirements are considered. (Kotonya and Sommerville (1998)).

### 5.1.1 Short introduction to requirements engineering

Requirements engineering studies the process involved in developing system requirements (Kotonya and Sommerville (1998)). Requirements engineering covers practically all of the activities involved in discovering, documenting, and maintenance a set of requirements for a computer-based system. According to literature, it has risen to be one of the most important, but difficult phases in product development. What is more, well-defined requirement and requirement processes in problem analysis and project management benefits the product development throughout the whole life-cycle from design to maintenance.

Despite the central role, there is still a lack of concrete evidence drawn from using systematic studies concerning the role of good requirements engineering. This was a task that Damian and Chisan (2006) set out to do recently with great results. Based on their findings, they suggest that an effective requirements engineering process at the beginning of a product development project had truly positive outcomes throughout the project life-cycle. Due to this, the efficacy of other project processes was improved ultimately leading to significant improvements in project negotiation, project planning, and managing feature creep, testing, defects, rework, and project

quality. Damian *et al.* (2005) have reached similar results. According to them, there lies a strong relationship between a well-defined requirement process and increased developer productivity, improved project planning through better estimations and enhanced ability for stakeholders to negotiate the project scope.

As always, the difficulty with measuring the effects of improved requirements engineering processes lies in the nature of the system as it did with the R&D processes too. It is hard to isolate the benefits from the whole system and it is also difficult to measure these benefits while the process is still in progress. One big problem is also the lack of historical data. So, when there is no data about the current situation, it is challenging to change the current processes. (Börjesson and Mathiassen (2004))

## 5.2 Software errors and their costs

Several studies have indicated that keeping errors in software unresolved is financially unbeneficial. Boehm (1981) states that the error correction in products that are on the field already, may cost over 1000 times more than the correction in the product requirements phase. Figure 10 illustrates how much the error correction does cost at each development phase, from requirements definition to field operation. Westland (2002) goes along with this statement. According to his results, uncorrected errors become exponentially more costly with each phase in which they stay unresolved. Perttula (2007) even suggests that in the mobile phone industry the cost of fixing errors in the field can become much higher because some mobile device models are being sold in quantities of tens of millions. Therefore he recommends to efficient error detection at on early point of development phases.

As it is known, numerous errors during the development process do not necessarily mean that the product quality is bad if the errors have simply been fixed. This goes *vice versa*; small amount of errors can mean that the testing has been executed poorly.

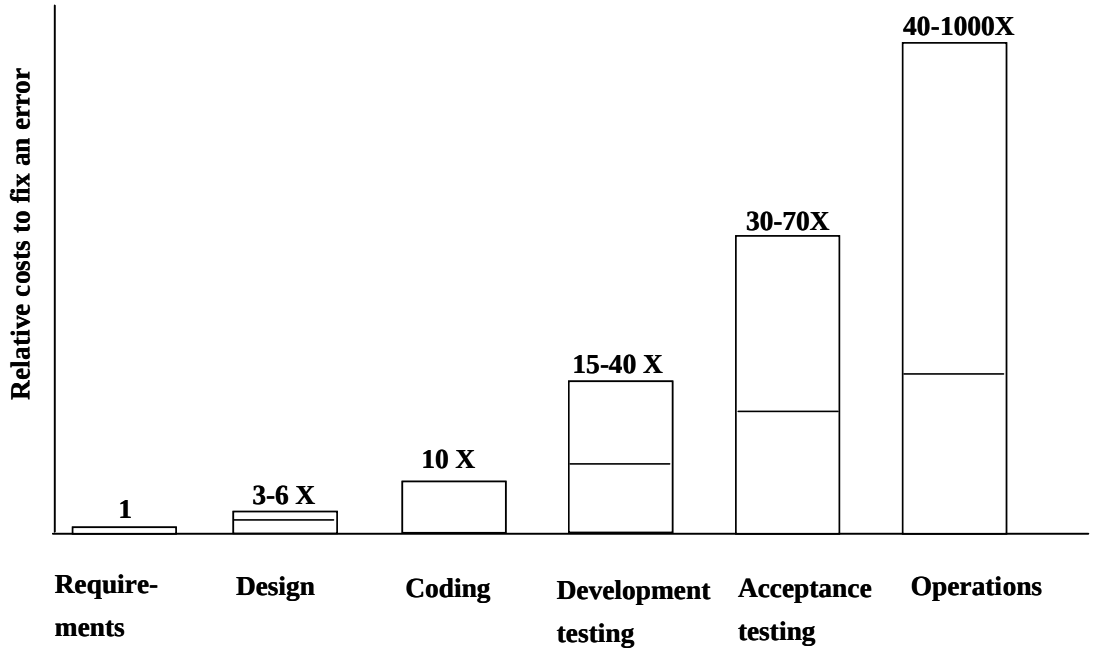


Figure 10: The cost of fixing errors increases when the product becomes readier. 63 projects were included in the analysis (adapted from Boehm (1981)).

## 5.3 Product life cycle time and slip

One of the main variables when measuring R&D is the product development time. Shorter time-to-market can speed up the revenue growth, at least if the competitors are not able to keep up the same pace. On the other hand, if the competitors are capable of creating shorter life-cycles, this will diminish the revenue of those not able to do it. Below, the most significant benefits of shorter time-to-market are depicted in a more detailed way. (McGrath (1996))

**Increased revenue**. Because of the shorter development time, the revenue will be increased throughout the entire product's life cycle. This can be seen more clearly from Figure 11. The normal life cycle time is represented in light gray. The product has been launched at the time of three years. After approximately two and a half years, the product has peaked after which a ramp-down has followed until the product has been terminated or replaced with a new and more attractive product. (McGrath (1996))

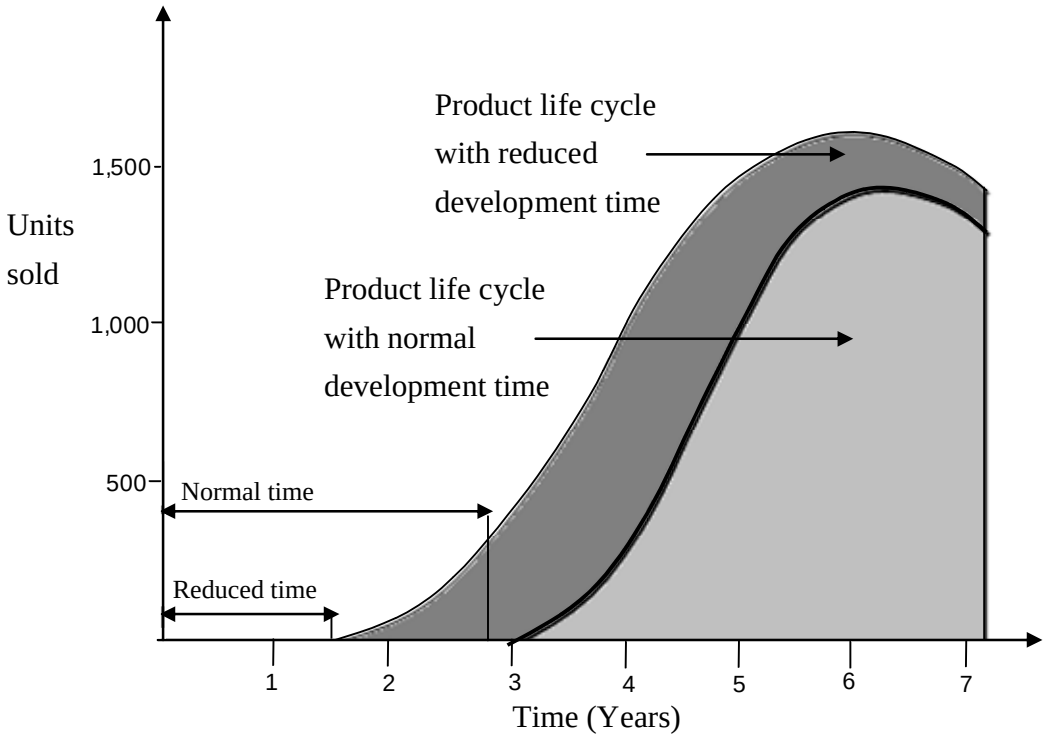


Figure 11: Product life-cycle curves with normal (light gray) and faster (dark grey) time-to-market. The picture has been adapted from McGrath (1996).

**Increased market penetration as a result of being first on the market.** When the product has been introduced first to the market, it has a great possibility to acquire a leadership position. This can be explained in three different ways. First, the company has been able to respond fastest to a new market opportunity. Second, the company is capable to be the first to apply a new technology and third the company has been able to respond the quickest to changes in the market. (McGrath (1996))

However, it is not enough to be first on the market. EMI for example was the first to develop the original CAT scanner, but because they did not have the support and service part handled well enough, its competitors GE and Technicon, who marveled in the sector, drove past EMI with their more complete package. (McGrath (1996))

**Success in time-sensitive markets.** In some industries, it is only possible to enter the market during a short time period. So, for the companies acting in such markets the time-to-market becomes extremely crucial. One example is customer specific components like custom semiconductor devices. The company has to be able to develop the component in time, because it needs to be designed early in the customer's end product. Otherwise the company does not get into the business but some competitor will. In such industries, time-to-market and predictability are key

competitive advantages. A good example is Sun Microsystems. In the 80's it was able to develop Sun 3 rather fast in the right spot. Their revenue almost ten folded in 3 years and their market share grew over 10%. (McGrath (1996)) In the mobile phone business there are also some recommendable times to launch a product. For example companies try to launch new products before Christmas season, since phones are popular presents.

**More successful products.** According to McGrath (1996) and Wheelwright and Clark (1992), the improvement in product development processes has a very big positive effect on the new products. When the product's development cycle is shorter, the conditions in the market do not yet vary too much. This can be interpreted in two ways. First, the consumers' taste does not change remarkably. If the cycle is too long, the "new" features may become outdated or just uninteresting. Second, when the cycle is long the competitors have time to introduce more innovative products. In short, the accuracy in estimating the future market conditions declines dramatically, the longer the development takes time.

**Lower development costs.** As illustrated in Figure 12, project costs have a strong correlation with product development costs. This has a logical explanation. Majority of the product development investments are run-rate based which means that a fixed number of employees work as long as the project is on. However, the project cost is not directly proportional to the cycle time because of some fixed costs like the price of equipment. McGrath (1996) suggests that a 50% reduction in cycle time typically leads to a direct reduction of development costs between 30% and 35%.

In addition to the already mentioned factors, there are some not so direct ones that are more difficult to measure. One is the effect on the brand. For example if the customers are not able to trust the launch date that the company has set, it will create a snowball effect that makes customer's products also slip. For these kinds of delays, there might be a penalty fee for the company.

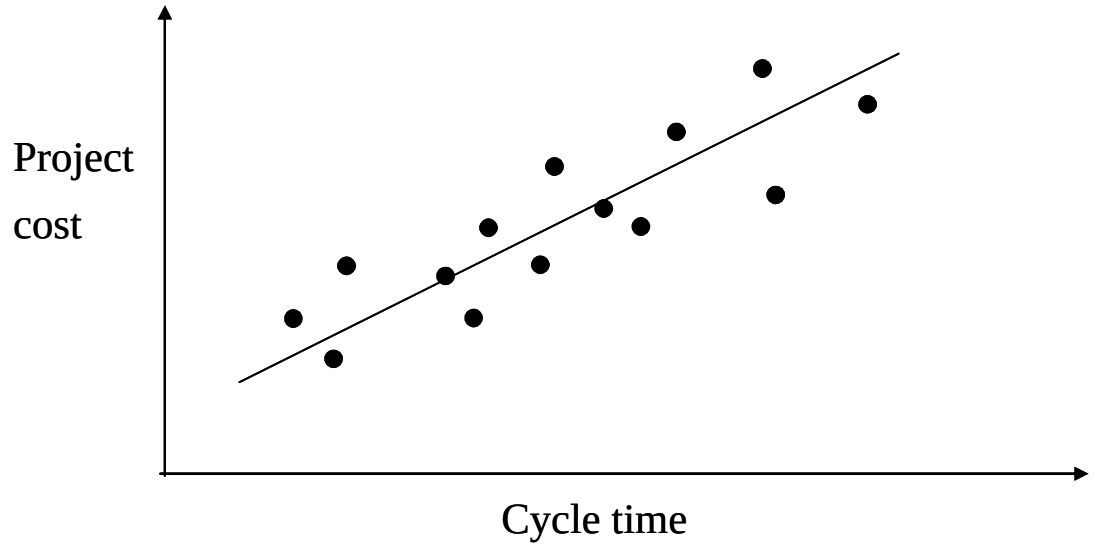


Figure 12: The relationship of product development cycle time and the overall project cost. The picture has been adapted from McGrath (1996).

Even though this chapter has set immense weight on time-to-market, it definitely is not everything. The world is full of projects that have slipped significantly, but after being introduced to the market, have proven to be hit products. However, nobody knows whether these projects would have been even more successful had they been on time.

## 5.4 Product quality

In practice a quality product means a consistently good product with a high reliability. One of the most important factors for the customers today is quality. The buyers do not want to worry whether the new product will work on the following day or not. Anyhow, quality is a rather subjective characteristic since not all customers have equally high quality standards.

Quality is thus a critical factor in high tech products that are expected to last at least a few years. In this thesis, the target was to include some quality variables in the model but unfortunately a suitable variable could not be found. The main problem was to find a variable that could be measured at each milestone of the product development process, but this kind of a variable proved impossible to find since the testing comes into the picture at the last milestones.

# 6. STATISTICAL METHODS

This chapter will go through the statistical methods used in this study. These methods can be considered to have evolved on top of each other. Basically PCA (principal component analysis) is the basis for PLS (partial least squares) and PLS again is the basis for the batch statistical process control (BSPC). Statistical process control (SPC) will also be covered, since the main idea of BSCP is greatly based on it. The chapter will end by describing how these methods have been applied to Nokia's product processes.

## 6.1 Principal component analysis

Principal component analysis (PCA) is an efficient technique to reduce multidimensional data into lower dimensions for analysis making. One of the main benefits is the way, how the data can be plotted in a low-dimensional plane to get a clear overview that can easily be interpreted compared to the original data that can be in tens of dimensions. The overview typically reveals groups of observations, trends and especially outliers that are difficult to detect otherwise. The plane can also help to identify any relationships between observations and variables and between different variables too. (Eriksson *et al.* (2001))

PCA has been applied in many fields from bioprocess engineering to functional neuroimaging. Basically, it can be easily applied to almost everything with multidimensional data. For example, Lintinen and Siimes (2004) have applied it to the visualization of the behavior of protocol stack processing. Eriksson *et al.* (2004) on the other hand have used it to visualize a different kind of data including genomics data, proteomics data and metabonics data.

The PCA data usually consists of an $N*K$ matrix where N represents the observations that are under study and K represents the variables that are available from the observations. In practice, the observation can be for example a chemical reaction where the variables provide the concentrations of the ingredients, pressures, temperatures etc. In this study, the observations are mobile phones and a variable is something that has been measured during the product development process. (Eriksson *et al.* (2001))

### 6.1.1 Pre-processing of the data

Before the multivariate data can be plotted, there is often some pre-processing of data involved. Pre-processing is extremely important since it can make the difference between the model in question being applicable or not. The two most common methods are *scaling* and *mean-centering*. Additional methods are also found such as *advanced scaling* and *data correction and compression* but they are not covered in this thesis. (Eriksson *et al.* (2001))

The key idea of *scaling* is to make all the variables of equal length. Different variables can have a huge variability inside the numerical range. Because the variables with a large range have also a large variance, they easily dominate the variables with smaller variance, thus it is important to normalize both the small and the large values in order to make the data equal (Figure 13). The most common scaling method is probably unit variance scaling. It calculates the standard deviation ($s_k$) for each variable. The scaling weight is then reached as the inverse standard deviation ($1/s_k$). After this, each column X (observations) is multiplied by the weight so that all columns have the same variance. (Eriksson *et al.* (2001))

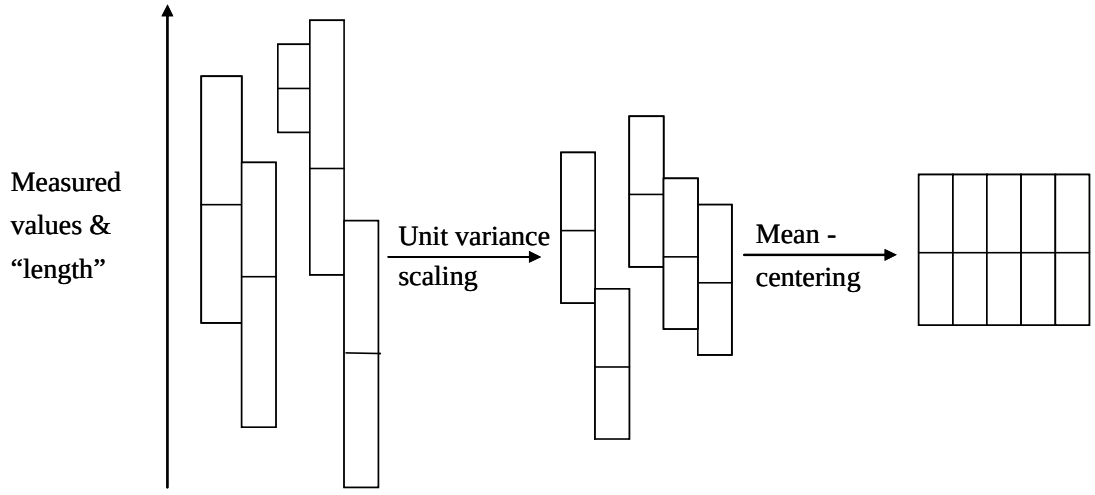


Figure 13: Scaling and mean-centering (adapted from Eriksson *et al.* (2001))

Like the name suggests, *mean-centering* calculates the average value of each variable and then subtracts that value from the data value, see Figure 13. (Eriksson *et al.* (2001))

### 6.1.2 A geometric interpretation of PCA

As mentioned before, a PCA data matrix normally consists of N observations and K variables. These variables can be presented in a variable space where each variable has its own dimension, that is, an own axis (Figure 14).

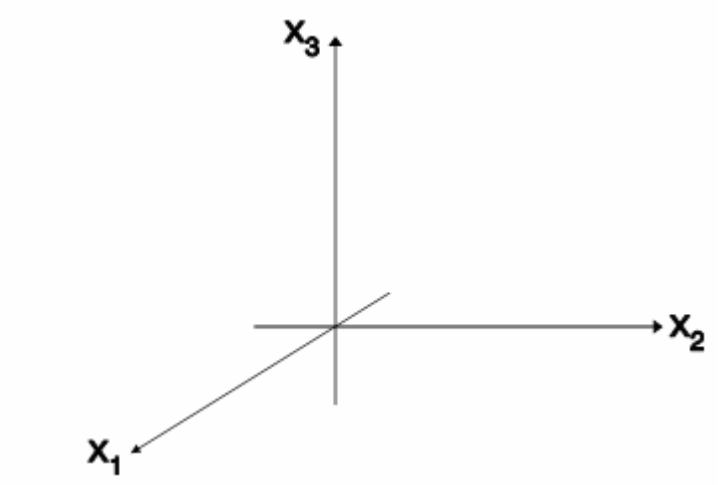


Figure 14: A variable space with three axis. (Eriksson *et al.* (2001))

When the observations are plotted, they are first situated like a cloud in the variable space. Due to the mean-centering, the average point and the cloud around it move to the origin (Figure 15).

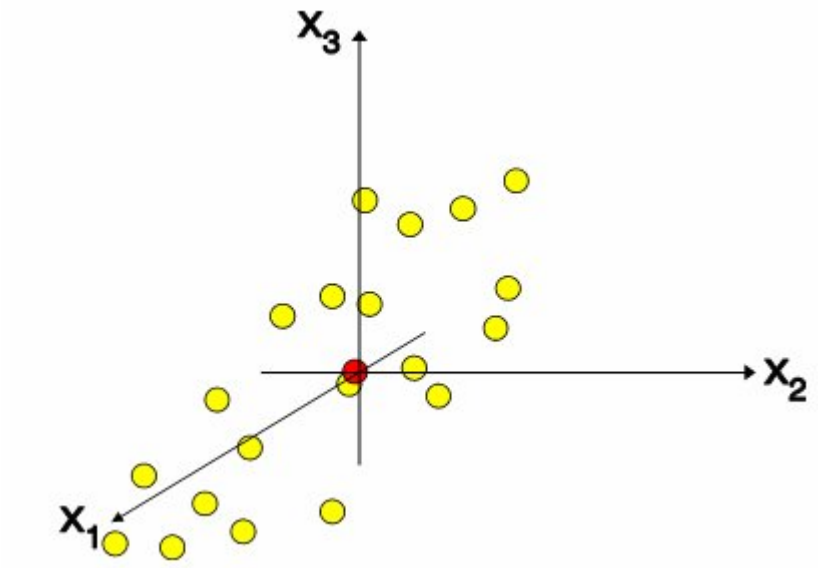


Figure 15: The observations are in a cloud around the average point after the mean-centering. (Eriksson *et al.* (2001))

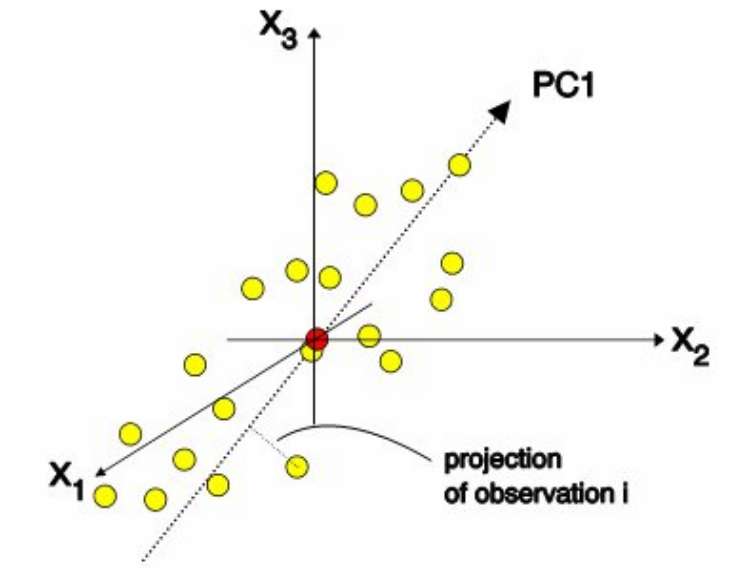


Figure 16: The first principal component. (Eriksson *et al.* (2001))

When the data has been scaled and mean-centered, the first principal component (PC1) is calculated. PC1 is fitted in the variable cloud using least square calculation. In practice, the line goes through the average point of the data and the variables can be projected onto the line. The resulting co-ordinate value along the line is called *score* (Figure 16). (Eriksson *et al.* (2001))

Since one component is often not enough to explain the variability in the data, new components are usually generated in the model. The second component (PC2) is orthogonal in relation to the first component and also cuts origin. Now, with the two principal components a plane is defined. As mentioned before, the co-ordinate values are called scores, and when the observations are projected in this plane, the resulting plot is called the score plot, see Figure 17.

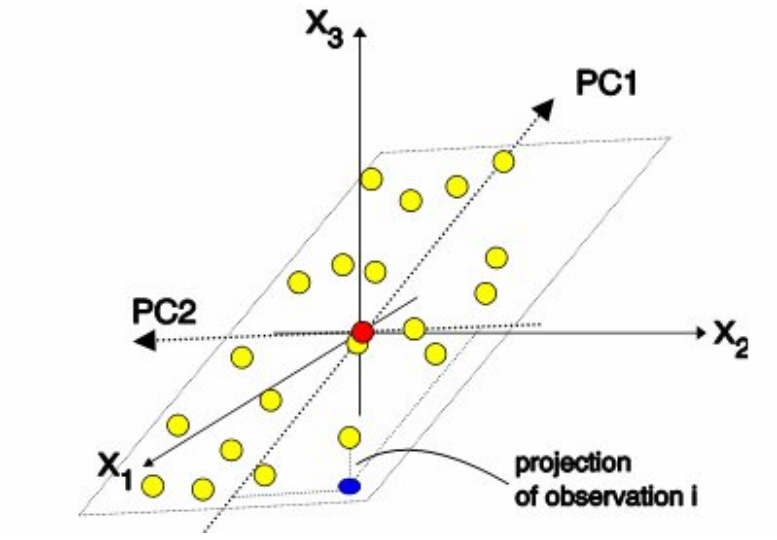


Figure 17: 2nd principal component and the plane. (Eriksson *et al.* (2001))

However, sometimes even two components are not enough to explain the variation in the data set. In this case, yet another component is added to the x-space. The third component, like the second, is also orthogonal to the first and the second component and also cuts the origin where the average lies. Thus it is the component that further explains the remaining variation in the data. Several additional components can be added with the same logic to the model. The equation of PCA can be formulated as following:

$$X = 1 * \bar{x}' + T * P' + E \qquad (1)$$

where X is the data table, $1 * \bar{x}'$ is the data after mean-centering and T*P' is the structure of the model, that is, the components (T) and the loadings (P) and E is the residual matrix. (Eriksson *et al.* (2001))

As mentioned before, PCA is especially useful in plotting the data. In practice, this means the score plots, where the components form the axis in question. If the plot shows something interesting e.g. outlier observations, the variables affecting this can be identified easily. The whole process has been described in brief in Figure 18.

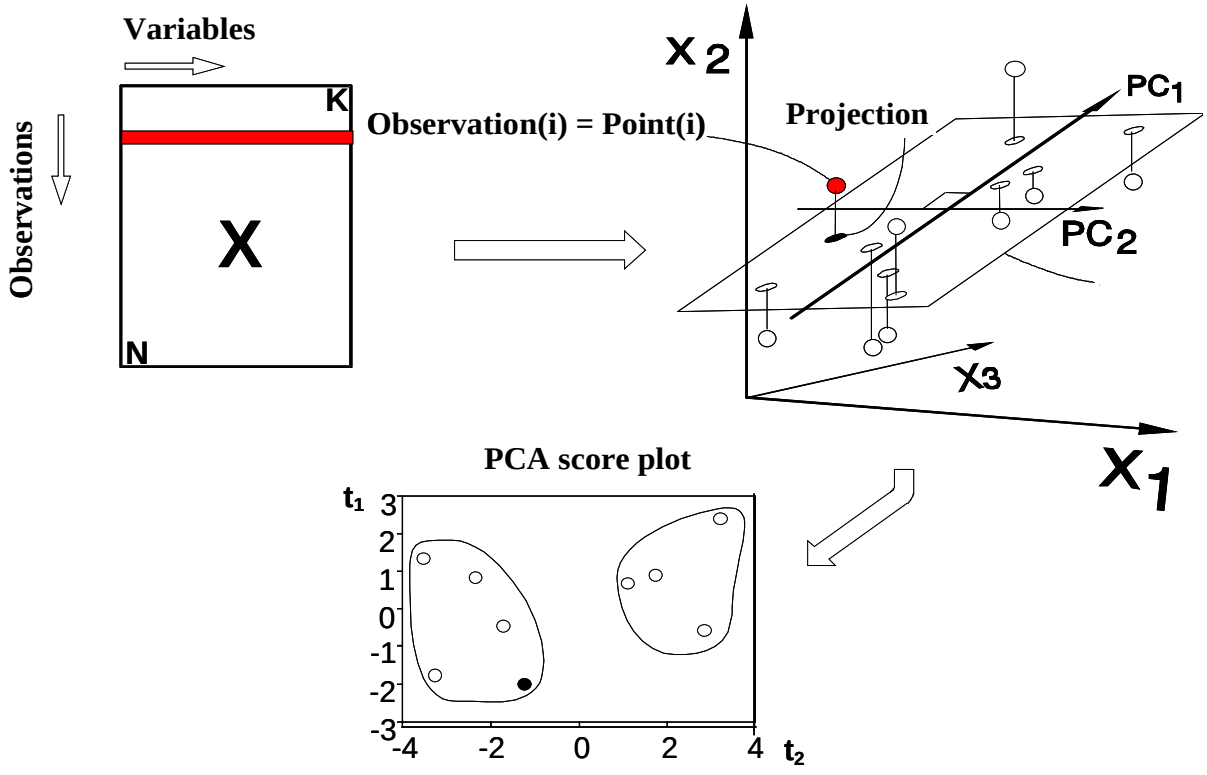


Figure 18: The PCA method in brief (adapted from Eriksson *et al.* (2001))

Most often the score plot with observations is compared with its loading plot. The loading plot depicts the variables that are influencing on the location of the observations in the score plot. In practice, this means that if the location of the variable is far from the observation and especially in diagonally opposed quadrants, the correlation between them is negative and if they are in the same spot, the correlation is positive. In addition, the distance of the variable from the origin is significant. The closer the variable is the origin, the smaller impact it has on the model and vice versa. (Eriksson *et al.* (2001))

### 6.1.3 What is the optimal number of components?

A key decision in PCA modeling is the number of components. The answer lays with the goodness of the least square fit ($R^2$) and the goodness of prediction ($Q^2$) based on internal cross validation. The $R^2$ can get close to one because if there are enough parameters there can easily be a good but random correlation with the model even though it would not be reasonable. This is why $Q^2$ is more important. Figure 19 depicts how the growing amount of components affects $R^2$ and $Q^2$. Practically, this means that when the prediction rate starts to decrease, the optimal component number has been reached. (Eriksson *et al.* (2001))

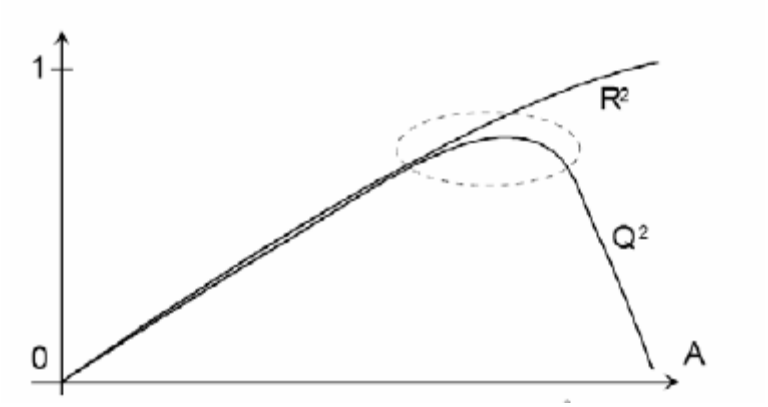


Figure 19: PCA component criteria (Eriksson *et al.* (2001))

## 6.2 Partial least squares

Eriksson *et al.* (2001) define partial least squares (PLS) as following: ”PLS is a regression extension of PCA, which is used to connect the information in two blocks of variables, X and Y, to each other.” PLS can also be defined as projections to latent structures by means of partial least squares. The three most common application areas for PLS are 1) quantitative structure-activity relationship modeling, 2)

multivariate calibration, and 3) process monitoring and optimization. One of the reasons why PLS is especially good is its ability to handle noisy and incomplete data.

Like PCA, PLS has been applied to many fields. For example Lintinen and Siimes (2004) have used it to model a processor workload. Kamis and Stohr (2004) on the other hand have modeled parametric search engines and the factors that make them effective when shopping online for differentiated products.

### 6.2.1 Geometric interpretation of PLS

As with PCA, PLS data (both factors X and responses Y) is in most cases scaled and mean-centered. If there is information available about the importance of the variables it can be used when weighting the variables. It can make the model remarkably better and more accurate. (Eriksson *et al.* (2001))

When PLS is put into a graphical form, the main difference with PCA is that instead of one point in a graph, there are two, one in X-space and one in Y-space. If there is only one Y-variable to predict, there is also only one axis in Y-space. When the amount of Y variables grows, it results that the amount of dimensions also grows respectively.

The components of PLS are calculated in a very similar way to PCA. The first PLS component is formed by a line in X-space and a line in Y-space (Figure 20). There are two additional factors for calculating these lines. Firstly, the lines must approximate the points well both in X and Y space. Secondly, the lines are situated in a way that they maximize the covariance between each other. It is noteworthy to mention, that the lines intersect with the average points that is the origin in both spaces. When the lines are defined, it is possible to calculate score vectors $t_1$ (X-space) and $u_1$ (Y-space), and their connection can be denoted as followed:

$$u_{i1} = t_{i1} + h_i \tag{2}$$

where $h_i$ is a residual. Whether the correlation between x's and y's is strong, can be easily visualized by a correlation plot, where $t_1$ is on x-axis and $u_1$ on y-axis. The straighter the line in the plot, the better the correlation is. (Eriksson *et al.* (2001))

As in PCA, new components are added as long as it increases both $Q^2$ and $R^2$. What

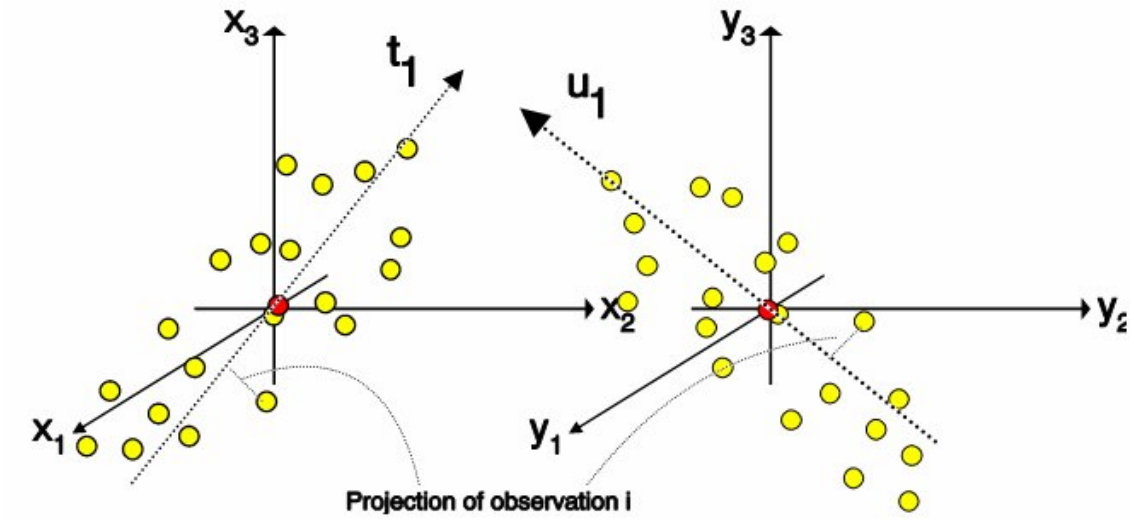


Figure 20: The first principle component of PLS. (Eriksson *et al.* (2001))

is important here is that the new lines improve the correlations between the X- and Y-planes. The general model of PLS can be written as:

$$X = 1\bar{x}' + TP' + E$$
$$Y = 1\bar{y}' + UC' + F \quad (= 1\bar{y}' + TC' + G) \qquad (3\&4)$$

where $1\bar{x}'$ and $1\bar{y}'$ are the variable averages that are the result of mean-centering. The information covering the observations is in matrices $T$ and $U$, and the information about variables is in matrices $P´$ and $C´$. $E$ and $F$ are the residual matrices. (Eriksson *et al.* 2001)

All in all, the main difference between PCA and PLS is that "the former is a maximum *variance* least squares projection of **X**, whereas the latter is a maximum *covariance* model of the relationship between **X** and **Y**". (Eriksson *et al.* (2001))

## 6.3 Statistical process control

Statistical process control (SPC), the forerunner of batch statistical process control, was originally developed by Walter A. Shewhart in the 1920's. SPC is considered to be an effective way to monitor the production process through the use of a control chart. Control chart, also known as *the Shewhart chart* or *process-behavior chart*, is one of the seven methods of quality control (along with the histogram, Pareto analysis, a check sheet, cause-and-effect analysis, stratification analysis, and a scatter diagram). The principle target of control chart is to indicate when the process is functioning as intended and when corrective action of some type is needed. Thus, in

the control chart the upper and lower tolerance limits are created around the average value and normally they are defined as ±3 standard deviations (Figure 21). The use of a control chart is quite simple. One has to first pick a variable that describes the process or the product well. Then during the process, this variable is plotted and if the value exceeds control limits, the cause is looked for and a corrective action is taken. Of course, sometimes the limits might be crossed without any serious reason,

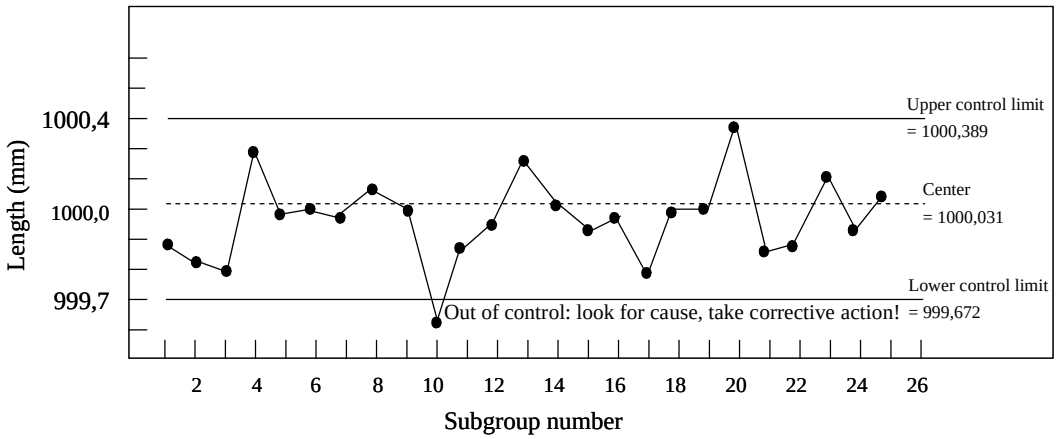


Figure 21: Control chart of a meter stick length (adapted from Kolarik (1995))

but since the limits are quite loose, usually the crossing does not happen for nothing. (Kolarik (1995))

The following chapter will concentrate on batch statistical process control (BSPC). This approach is very similar with SPC. The key difference, however, is that in SPC only one variable is under study in each plot, whereas in BSPC the “variable” is a score vector, which is a linear combination of several variables. In short, this means that the contribution of number of variables and their correlation with each other can be observed in only one plot.

## 6.4 Batch statistical process control

A typical feature for a batch process is duration, since with a distinction to continuous processes, batch processes have a finite duration. Batch processes can be found everywhere from the pharmaceutical sector to manufacturing of paints and inks. It has also been estimated that 75% of industrial processes would be batch processes.[15] The great significance of a batch process arises from the fact that thanks to batch modeling, these processes can be monitored easily during the process which

[15] http://en.wikipedia.org/wiki/Batch_production, accessed 30.10.2007

is not necessarily an easy task. This way it is possible to influence the process in question before problems grow immense.

There are many example studies of batch statistical processing. For example Antti *et al.* (2002) used batch statistical processing to $^{1}$H NMR-derived urinary spectral data. They created a model to define the mean urine profile for hydrazine-treated animals where each animal was considered as an individual batch. The batch models have been mainly implemented to bioprocesses and the standpoint where batch modeling has been used in this study is completely new.

A batch process can be considered to be an extension from PLS just as PLS was an extension from PCA. This is due to the fact that in PLS there were only one process with certain factors X and responses Y. However, in batch processes several similar processes are compared with each other. In addition, batch processes can have initial conditions, Z. This is illustrated in Figure 22. (Eriksson *et al.* (2001))

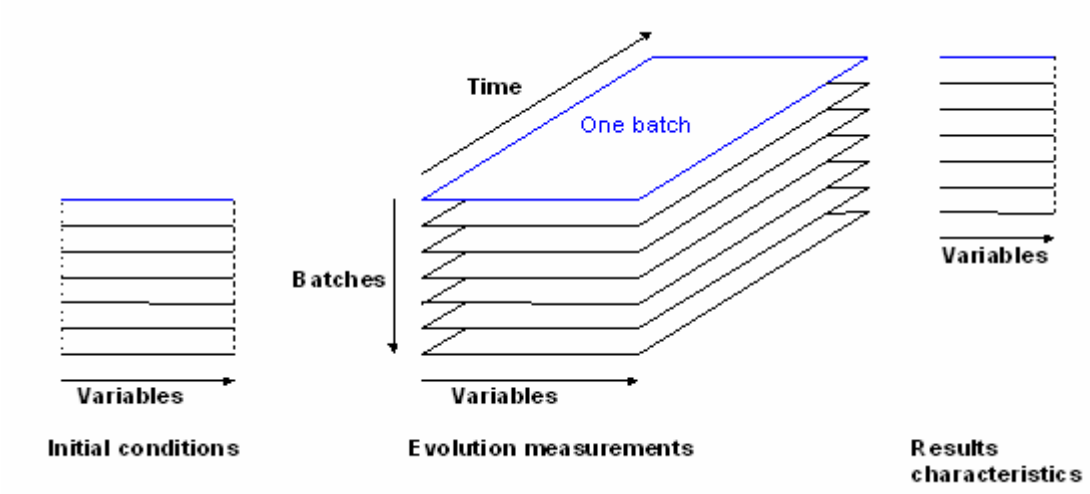


Figure 22: The main idea behind batch modeling. (Eriksson *et al.* (2001))

There are two main approaches to batch modeling, *the observation level* and *the batch level*. The observation level concentrates on evaluating an individual observation at certain points, predicting batch maturity, and understanding the evolution of a typical, good batch. According to the normal path of a batch, the new batches can be monitored later on. Thus observation level is especially good for fault detection. The batch level, on the other hand, concentrates on making a model over the whole batch which can then be used for classifying new batches as good or bad and to be able to monitor how e.g. initial conditions influence the outcome, Y. The batch level is out of the scope of this thesis, but nothing prevents its use in this kind of a setup. (Eriksson *et al.* 2001)

### 6.4.1 Modeling batch evolution

The observation level modeling starts with unfolding the data. The data has to be in a form where in each row there is one time point of a batch and the variables are listed in columns. Y can be either an already measured time value or then it can be a maturity index generated by SIMCA-P+ 11.5. From the received data, a reference PLS-model of X's versus Y's is created. At this point, the batches are put in a line in relation to time and an average value and a standard deviation is calculated, as illustrated in Figure 23. This way an average trajectory is reached with the upper and lower control limits created from the standard deviation. A contribution plot can also be generated from the data. This plot shows how much a certain batch has differed from an average or any other graph and whether this differing has been positive or negative.

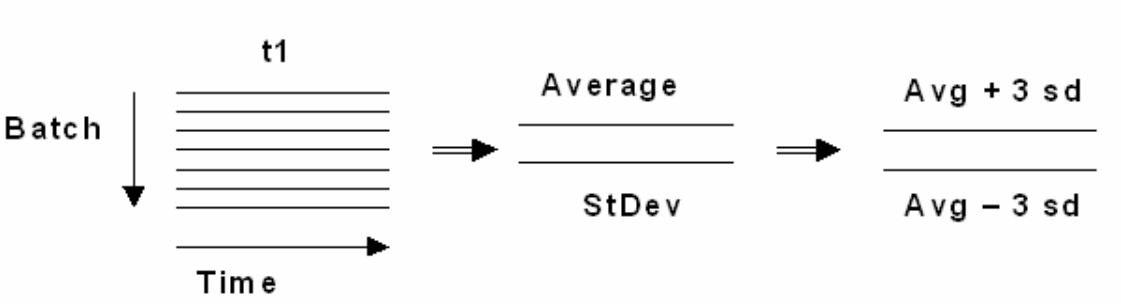


Figure 23: The theory behind the batch model boundaries. (Eriksson *et al.* (2001))

## 6.5 The batch modeling applied to Nokia's product process development

In this study, batch modeling was used to describe Nokia's product process development in order to monitor it and to find out if there are common patterns between the development processes of different mobile devices. The target was also to find out whether there were any product programs that differ from the mass.

### 6.5.1 The modeled data

In the modeling part, 64 different products were used. From these products 46 had reached their M4, that is launch, and 18 were still in the middle of the product development process. From these different programs the same data was collected at each milestone, from M0 to M5. The products came from all the three business groups (M, MP and ES) of Nokia. 35 of the products used S40 software and 29 S60 software.

The data consisted of requirement, error and time related data. The requirements were functional requirements and they were divided in three categories based on who was responsible for them. These categories were 1) the program itself, 2) other programs and 3) platforms. At each milestone, the status of the requirements was defined. Basically, this means that if a new requirement was defined to a product during the last milestone period (e.g. between milestone dates M2 and M3), it was considered as *new*. On the other hand, if a requirement was finished during the last milestone period, it was considered as *handled*. Again, if the requirement had received a "new" status at some point and had not yet been handled, it was considered as *open*. In addition, the total amount of new, open and handled requirements was included in the model as well as an open-% variable that was calculated using the formula (5):

$$open - \% = \frac{open\ requirements\ at\ milestone\ X}{cumulative\ amount\ of\ \ new\ requirements\ at\ milestone\ X} \quad (5)$$

The error data was constructed the same way as the requirements data. It was likewise categorized to three different classes. The classes were 1) platforms, 2) programs and 3) "not defined" depending on who was responsible of correcting the error. Since there were few errors that had not been classified yet, they were situated under the "not defined" class. For errors, there was also a categorization to detected, handled and open errors.

There were two variables related to time. The first was milestone time in days, which was basically the time between two milestones. The second time variable was milestone slip in days which means the difference of the actual milestone date and the planned milestone date. In total, there were 28 observations for 64 products in 6 different milestones.

## 6.6 The modeling

The modeling was carried out using the program Simca-P+ 11.5. The data was pre-processed using unit variance scaling and mean-centering methods. First, some PCA-plots were created. The same data was grouped in two different ways. In the first set there were as many observations as there were products. This means that the data of each milestone of a product, is in one dot in PCA-plot. In the second set, there were $N_{milestone}*N_{products}$ observations that is, every product had its own observations at each

milestone and thus every product has 6 spots in the PCA-plot. The target was to visualize the data in order to understand it better.

PLS modeling was also carried out. The objective was to explain and predict the product development time of the next milestone based on the information that had been received before that milestone. For example the time between the milestones M2-M3 was predicted using the information received before the M2 date.

Pre-processing was done with batch data too. The ideal amount of components was then selected and the batch model was created. From the batch model several plots were studied including score plots, variable plots, Hotelling's T2Range plot and Observed vs. time/maturity plot. After screening these plots, it became obvious that there were a few significantly differing products which were then removed from the model since they did not follow the "normal" product process path and were thus widening the batch standard deviation limits too much.

# 7. RESULTS

This chapter covers the results of PCA, PLS and batch modeling. PCA models mainly gave support for the validation of existing ideas for example that each milestone is an individual entity that has a special nature of its own. PLS model was, on the other hand, a disappointment since it did not give enough accurate results.

The batch model is the key model in this thesis, since it is the only model that can be used in product development regularly as an indicator of a product development process status. Thus, the interview results that end this chapter are completely concentrated on the batch model.

## 7.1 PCA plots

As mentioned in the modeling chapter the PCA plotting was carried out with two different data sets that were containing the same data but were organized differently. First, the model where there were as many observations as there were products is coverd and then the model with $N_{milestone}*N_{products}$ observations will follow.

### 7.1.1 PCA plots with the first data set

The first data set was colored according to the three business groups (Mobile Phones, Multimedia, and Enterprise Solutions) that develop mobile phones at Nokia. The resulting graph is illustrated in Figure 24. As the figure suggests, the three business groups are developing mobile phones a bit differently, what comes to the time, requirements, and errors.

However, this is quite intuitive since the complexity of the products is rather different between the business groups. It is only logical that a basic phone from Mobile Phones, which does not have e.g. any camera, Internet or radio would be easier to construct than Multimedia's complex devices. The produced quantities are also rather different. In the last quarter Nokia sold over 110 million mobile devices out of which Mobile Phones produced ~100 million, Multimedia ~10 million and Enterprise Solutions ~2 million devices[16]. This suggests too that the contents of MP are optimized to few features in order to be able to manufacture them fast and efficiently.

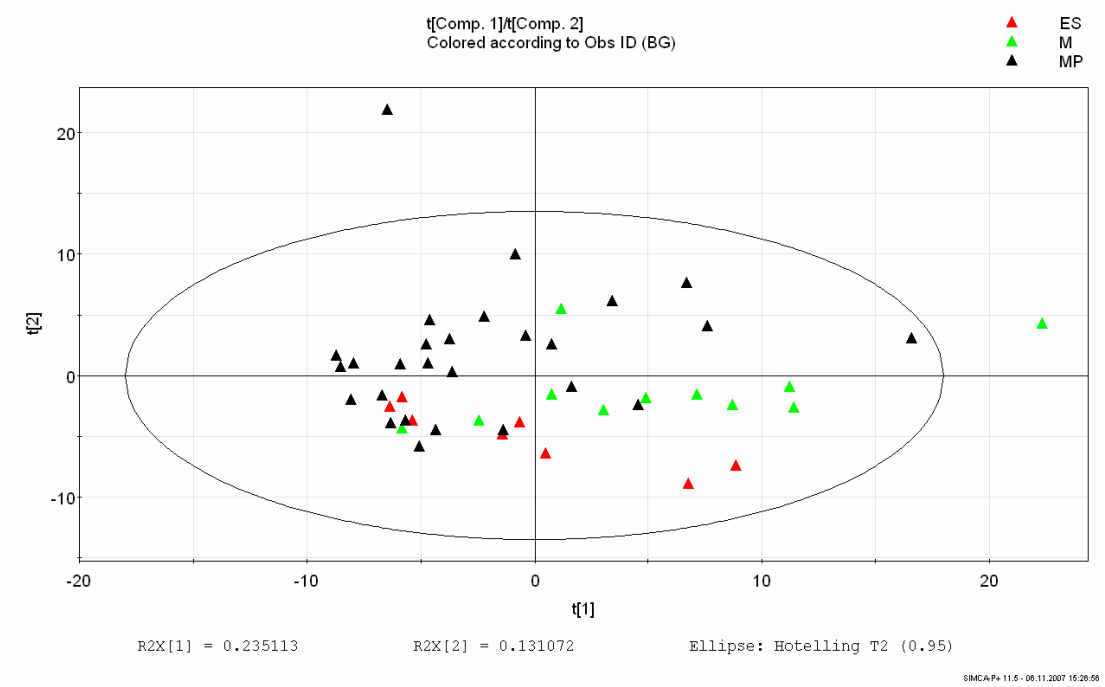


Figure 24: PCA model of different mobile phones colored according to the business group that has created the device in question.

In Figure 25, the mobile devices have been colored according to the software platform Series 40 or Series 60 that the phone is using. As the figure shows, the difference is quite visible and the points create separate swarms. The main difference between these software platforms is that Series 40 is designed for simple phones and Series 60 is designed for smartphones that have to use several different applications smoothly together. In other words, Series 40 is not designed for multitasking. However, since it is possible to do quite demanding devices with Series 40 and rather simple products with Series 60, the groups are overlapping each other slightly as expected.

In Figure 26, the variables behind the PCA plots (Figures 24 and 25) are illustrated. The variables are now colored according to the variable type. Requirement variables are plotted in green, error variables are plotted in red and the time data in different colors depending on what is the milestone. The graph illustrates that the requirement data and error data also form their own point swarms, which means that they have a bit opposite effects to the observations. When this plot is compared to the Series 40/Series 60 plot, it seems that the requirements affect the Series 40 devices more and the errors have a greater effect on Series 60 devices. This can be explained with

[16] http://www.nokia.com/results/results2007Q3e.pdf, accessed 30.10.2007

the fact that Series 40 software is simpler and more mature than the newer Series 60 software.

If Figures 24 and 25 are now compared with each other, it can be seen that the software platforms go hand in hand with the business group division as mentioned. The devices produced by MP are in the same section as Series 40 software platform and the other business groups M and ES are situated in the same section with Series 60 software platform. Again, the result is rather intuitive, since MP is the producer of mass-customized phones and M and ES produce more complicated devices for more demanding customers.

Because of the considerable difference between Series 40 and Series 60 softwares, these groups were modeled separately in the batch modeling phase. The same thing could have been done also according to the different business groups, but in that case, the observation group would have been too small and thus the results less reliable.

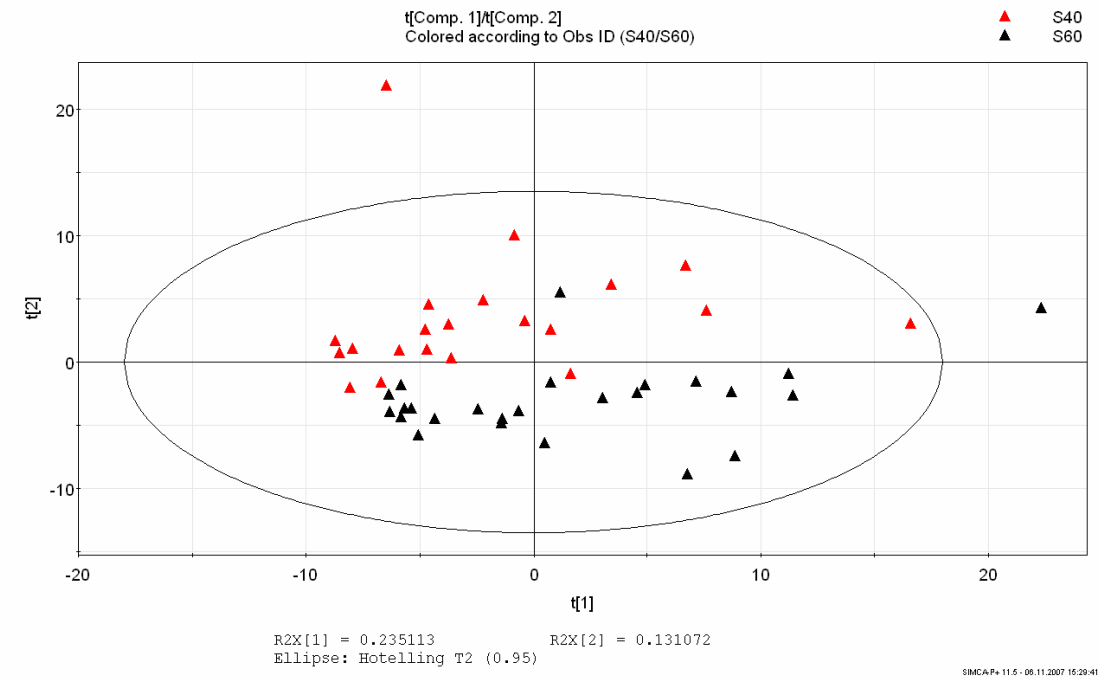


Figure 25: PCA model of different mobile phones colored according to what software platforms the mobile devices are using.

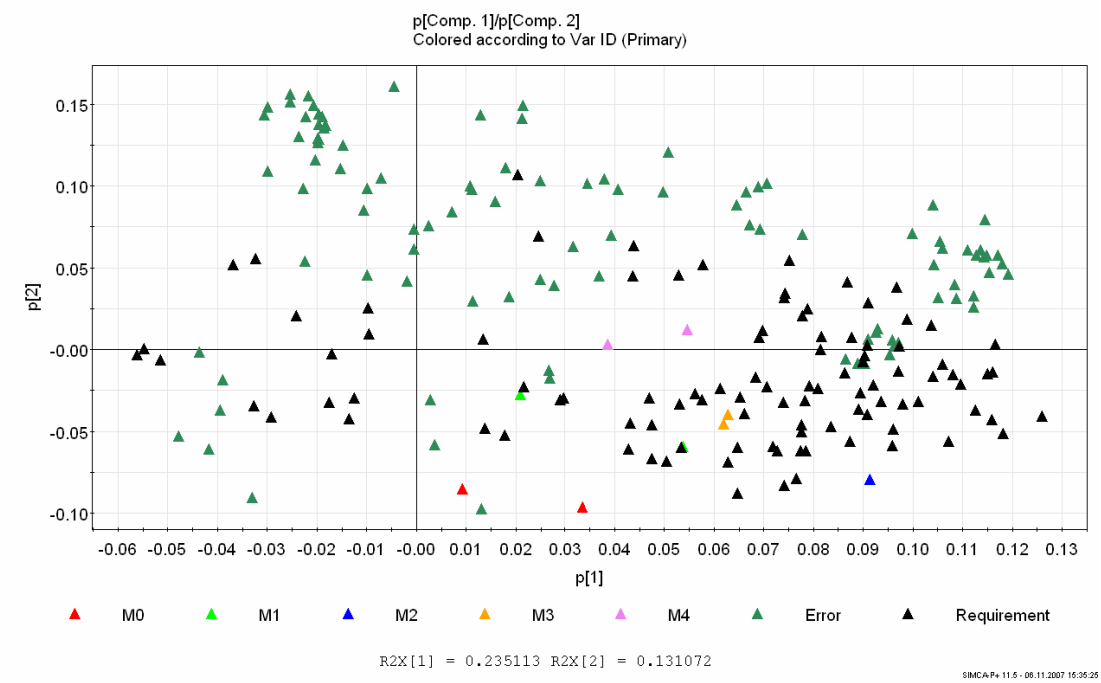


Figure 26: The variables behind PCA models.

### 7.1.2 PCA plots with the second dataset

A PCA score plot of the components (first and second in the plot) was drawn based on the data and the products were colored according to the milestone where they currently were (Figure 27). The graph shows clearly that each milestone forms a swarm of dots and these swarms slide from right to left showing clear “evolution”. This means that in each milestone there are variables that are more important in that particular phase than in the others.

The variables behind the model are plotted in Figure 28. When Figures 27 and 28 are compared, it is easy to notice that in the first milestones variables concerning new and open requirements are the most significant. In later milestones, the important variables are related to handled requirements and finally errors. The variables related to time, that is milestone time and slip, are quite in the middle of the graph suggesting that time and slip are equally important at each milestone.

The interpretation of the variables makes a lot of sense, since according to the CE process model during the first milestones the requirements are defined and they reach the “open” status. After this, the handling of the requirements starts and errors also

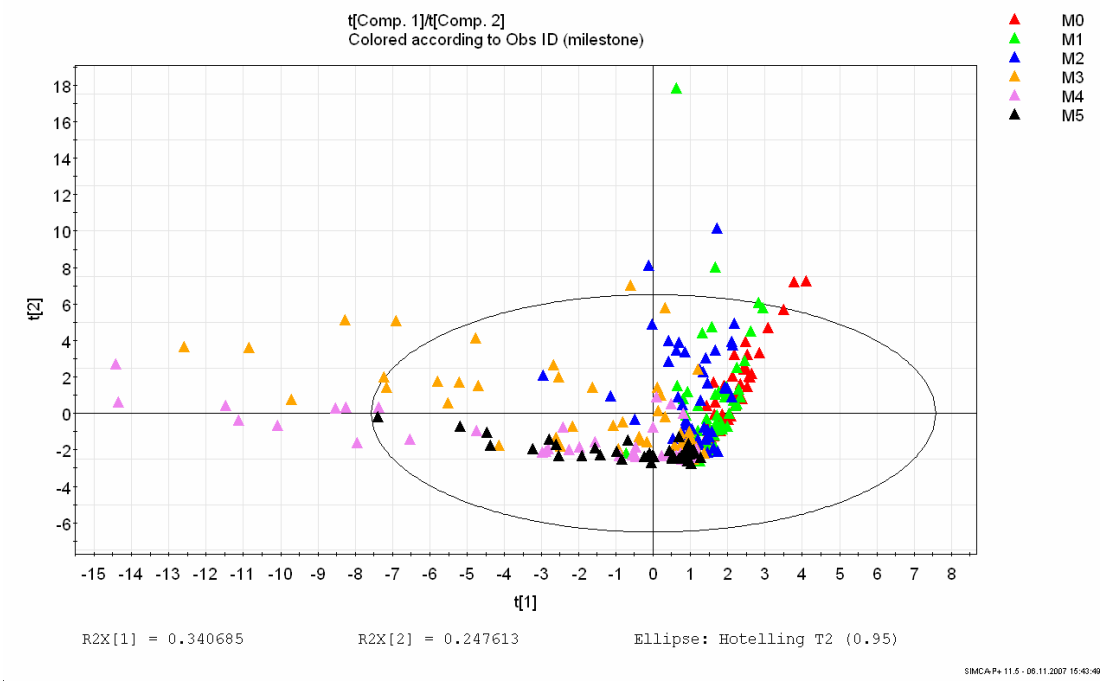


Figure 27: PCA score plot of products at different milestones

are detected and finally corrected. If the process was more like the iterative or the agile development model, requirements and errors would be emerging in a constant flow during the whole development process.

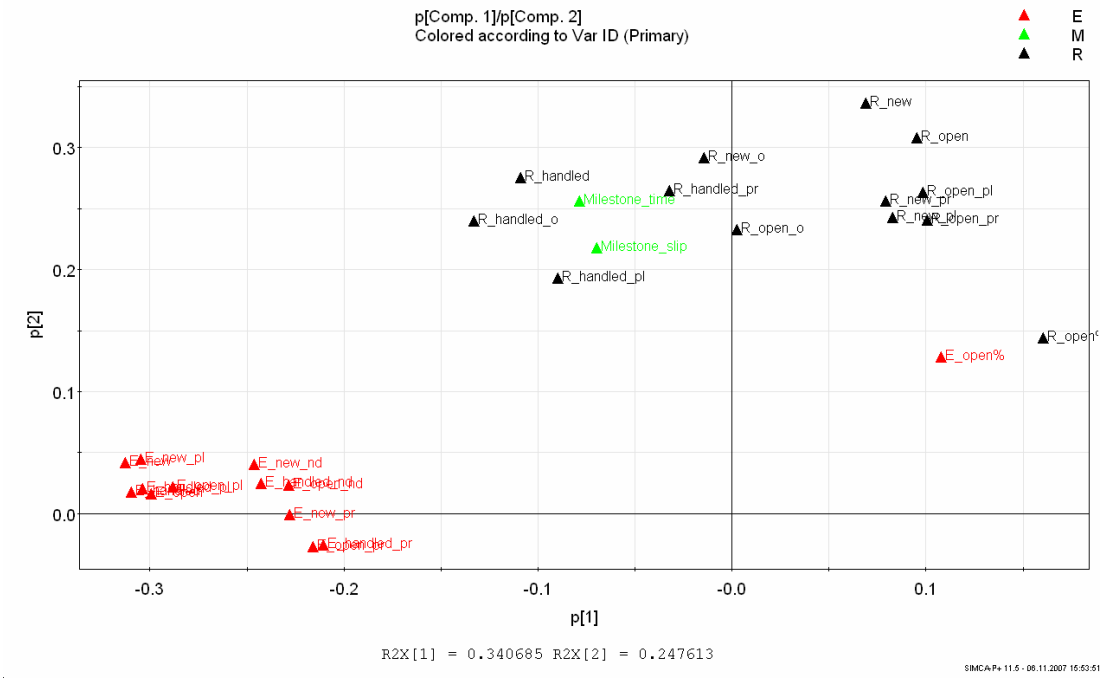


Figure 28: PCA loading plot of the variables

## 7.2 PLS results

The time prediction of each milestone did not work out as expected. The prediction accuracy $Q^2$ was at most 50 % and $R^2Y$ was only little better (Table 3). In addition, the RMSEE's (Root Mean Square of Error Estimation) was getting far too big values. The RMSEE was sometimes even over 25% of the products real development time. Thus, the estimates were too wide to be useful. What is more, when the models were validated, it turned out that every one of the models was too unreliable confirming that they should not be used in real life.

It is, however, quite logical that the predictions did not come true, since there are many factors that were not taken in account in this modeling. These include for example the workforce and overtime work. If a deadline is close, more workforce can be added in order to complete the project on time. The model does not cover quality either. Are the products on time, but the quality is bad? Or vice versa, has the product been late because there were especially high quality targets. In short, the process variables that have been used in the model do not cover all the information related to the development time.

Table 3: The key parameters for the PLS models at each milestone. M5 could not been explained with the data. For M0, part of the data was unavailable.

| | R2Y | Q2 |
|---|---|---|
| **M5** | 0 | 0 |
| **M4** | 0,42 | 0,28 |
| **M3** | 0,61 | 0,53 |
| **M2** | 0,33 | 0,19 |
| **M1** | 0,37 | 0,13 |
| **M0** | - | - |

## 7.3 Batch processing results

### 7.3.1 The batch model

As already mentioned in R&D process models Chapter 5, there are different stages when creating new products and there are different targets at each milestone. This suggests that different variables are important at different milestones. To give an example, after the launch there cannot be any new requirements anymore, since what is the use to create new requirements for a product that is already on the market. In

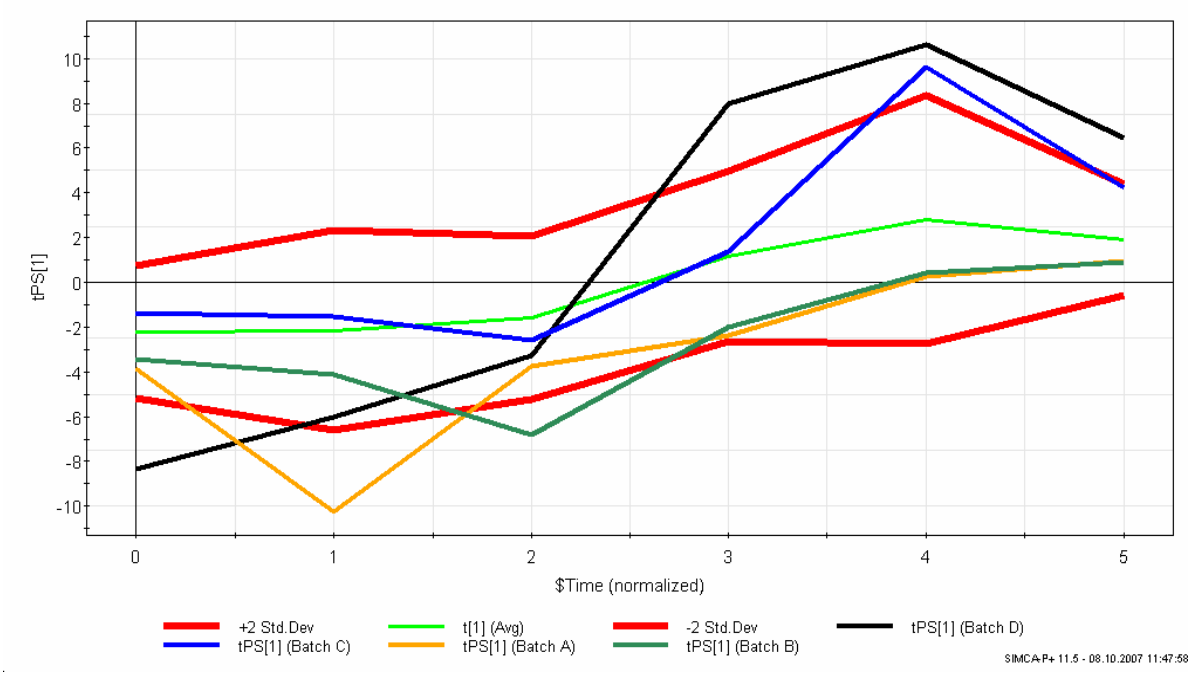


Figure 29: The resulting batch model. In y-axis is the linear combination of all variables and the x-axis represents all the milestones, M0-M5.

addition, there cannot be any errors before the implementation of requirements has been started, because the errors arise from the requirements that have not been implemented in a correct way. The PCA graph with milestone coloring gave also support to this idea.

The final batch model with few outliner products (A, B, C, D) can be seen in Figure 29, where the first component of the model has been plotted. In X-axis there are the milestones from M0 to M5 and in the Y-axis there is the linear combination of the all variables. In this model, there are altogether 4 principal components and the contribution of each of them to $Q^2$ and $R^2Y$ of the model has been illustrated in Figure 30.

From the shape of the graphs in Figure 29, nothing definite can be said, but because different milestones have own characteristics as it was described in chapter 4 and illustrated in PCA-plot (Figure 27), it can be suggested that the products A, B and C have had problems with requirements and products A and D have had problems with errors. In the Chapter 7.3.3. the reasons for crossing the control limit lines are studied in detail. It confirms for one that during the first milestones the products cross limit lines due to the requirement related problems and during the latter milestones products cross the limit lines due to the errors in the products. This information

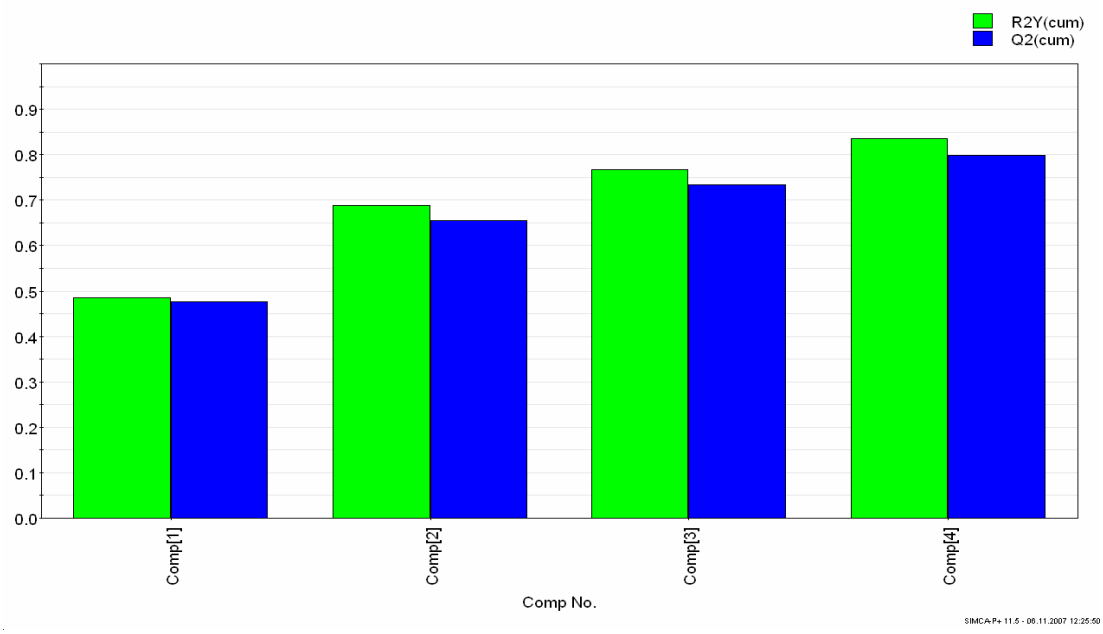


Figure 30: The four principal components with $R^2Y$ and $Q^2$ values.

comes from the contribution plots where average products and the outliners are compared with each other.

Simca's internal model validation was also used and the results from that are shown in Figure 31. In the validation, the positions of Y variables are changed randomly while the values itself are still the same. After this $R^2Y$ and $Q^2$ are calculated with the new data set. This procedure is then repeated a chosen number of times and the

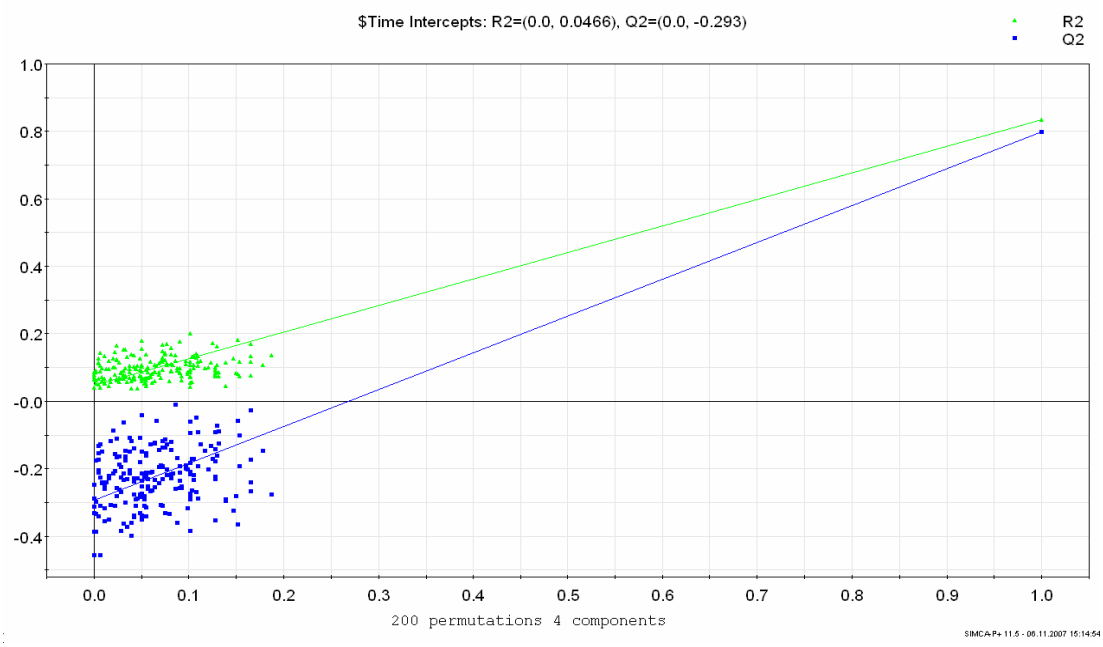


Figure 31: The validation of the four component batch model.

result plotted. As long as the new $R^2Y$-intercept is under 0.3-0.4 and $Q^2$-intercept is lower than 0.05, the model is statistically significant. (Eriksson *et al.* 2001) In this validation 200 permutations were used to see whether the model was reliable. As can be seen from the plot, it is highly unlikely that the same results would be achieved randomly, so the model used is rather trustworthy.

### 7.3.2 Spotting the outliers

Several methods were used to spot the outliers between the products. The simplest of them was to check each of the component plots to see whether any of the products were crossing the standard deviation limits. The first component with its control limit lines is illustrated in Figure 29 with some problematic products. The subsequent components were also viewed through, but they did not give any additional information about the products and were mainly highlighting the same problem candidates that were already visible in the first component Figure 29.

Simca also provides other methods for spotting the outliers. Hotelling's T2-test[17] with the 4 unusual products is illustrated in Figure 33. The results are the same as in the component plot. Products A and B are clearly out of the 99% control limit line and the products C and D clearly cross the 95% control limit line.

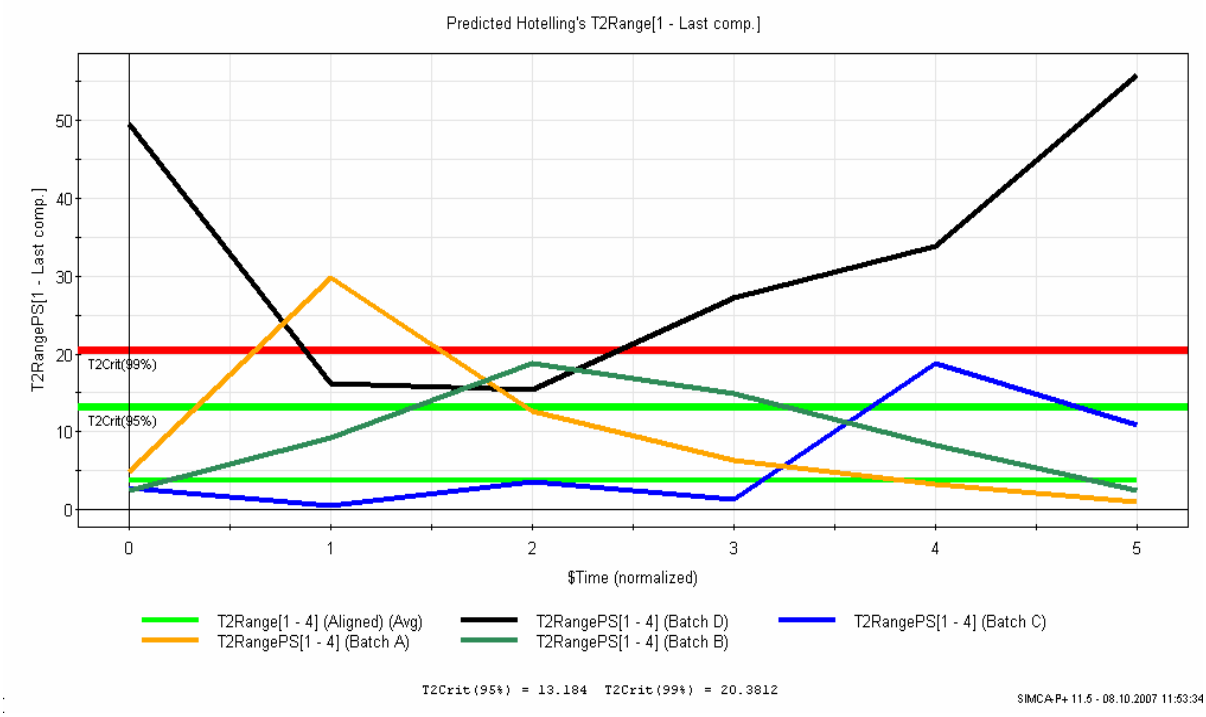


Figure 33: Hotelling's T2 test and the outliers.

[17] Hotelling's T2 test is the multivariate extension of the well-known Students's t-test. In Student's t-test, differences in the average response between two data sets are studied. T2 is used when the number of response variables is greater than one.

Based on the component plots and Hotelling's T2 test, product A was decided to be left out of the model. This was due to the fact that it widened the control limits too much allowing too many products to fit between the limit lines. It was also known that product A had an exceptional product creation life, so the omission was justifiable also from that point of view.

### 7.3.3 The problematic products

Component plots and Hotelling's T2 test identify well the outliers, the problematic products, but they do not tell the reason for this unusual behavior. To clear this out, variable contribution plots were drawn. These plots show the relative difference of the average program and the program that is under study and it is weighted with the importance of the variables.

In Figure 34, the contribution plot of product A at milestone M1 is presented. The figure illustrates that A has remarkably more new requirements (R_new) than the average program. These requirements are mainly platform requirements (R_new_pl) but there are also numerous requirements copied from other programs (R_new_pr) and also "own" requirements (R_new_o). This suggests that product A has been dependent on others which raises the risk of slipping because if the platform is late, this slip will cumulate on the product too. Figure 354 also suggests that other programs have completed quite a many requirements (R_handled_pr), but despite of this the platform and the other programs have left many requirements open (R_open_pr, R_open_pl).

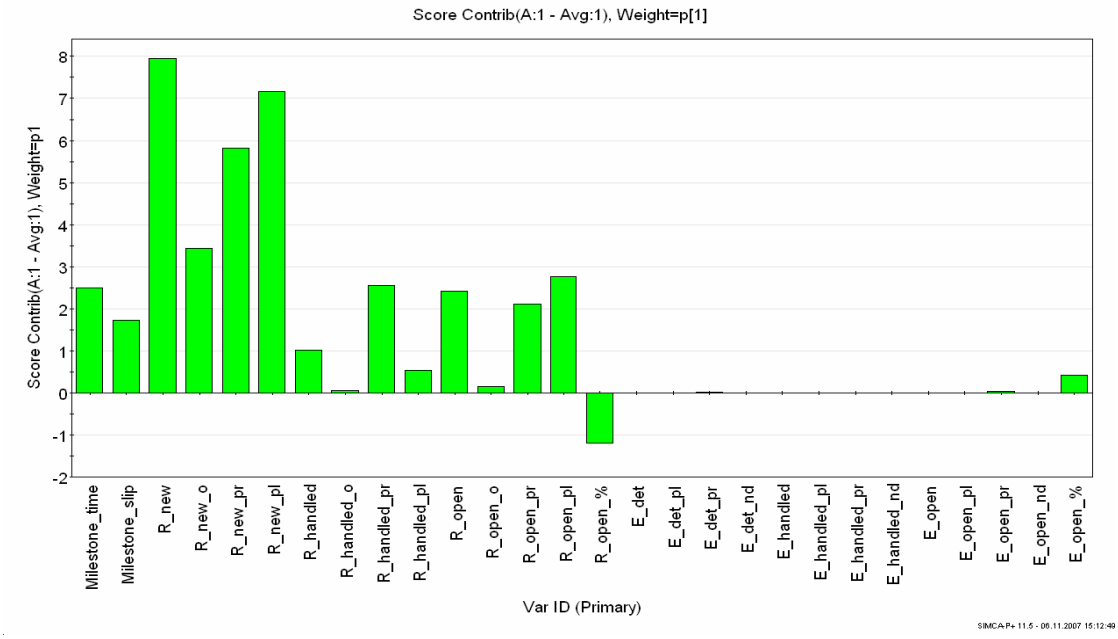


Figure 34: The contribution plot of product A at M1.

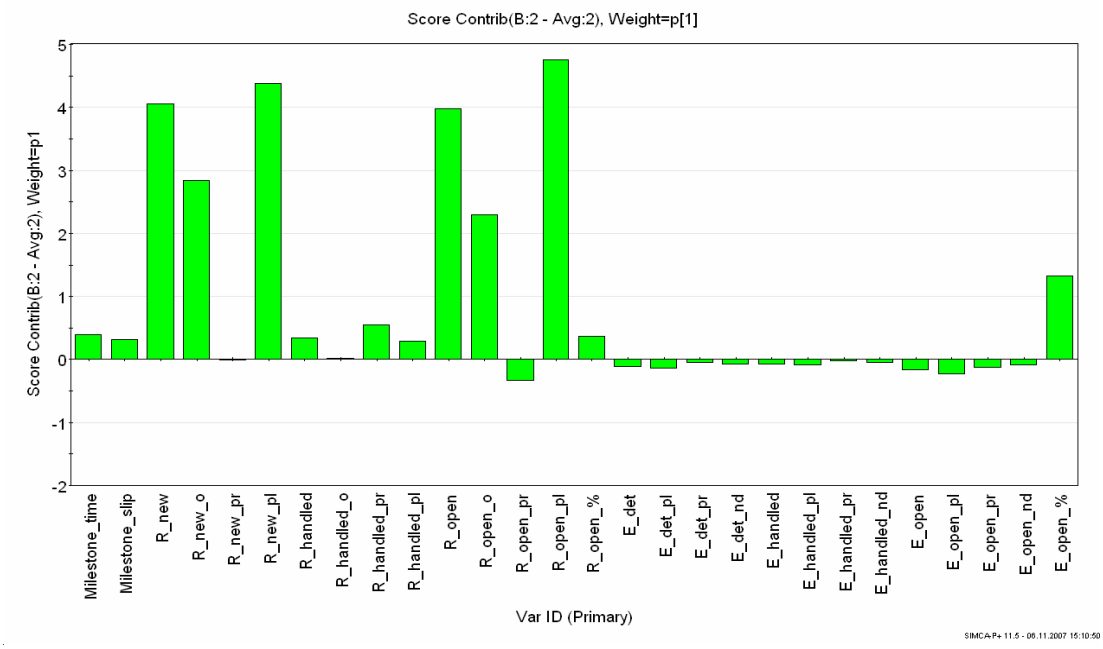


Figure 35: The contribution plot of product B at M1.

Figure 35 shows a similar situation for product B at M2. There are more than average of new requirements and the responsibles are the product itself and the platform. Again, a great deal of requirements is left open. What is noteworthy, is that the product has reached M2 but many requirements are still open. If the process model were following the agile or the iterative process model, it would be normal to have new requirements practically all the time, but this is not the case with the CE process model.

The Figure 36 reveals a different scenario. Program C has crossed the standard deviation limit lines because of the great number of errors before M4 (E_new). These errors are mainly in the product (E_new_pr), not in the platform (E_new_pl), but the platform includes a great deal of them too. What is interesting, is that product has corrected a quite many of errors but still some of them has been left open. However, it should be kept in mind that the amount of errors does not necessarily mean a bad product but efficient testing and a small amount of errors can signal that the errors have simply not been found.

### 7.3.4 The simulation of product C

Since late error handling increases the costs of the product significantly, it was decided to perform a simulation with test data derived from the error data of product C. This was to see, what kind of impact the different error handling has to the

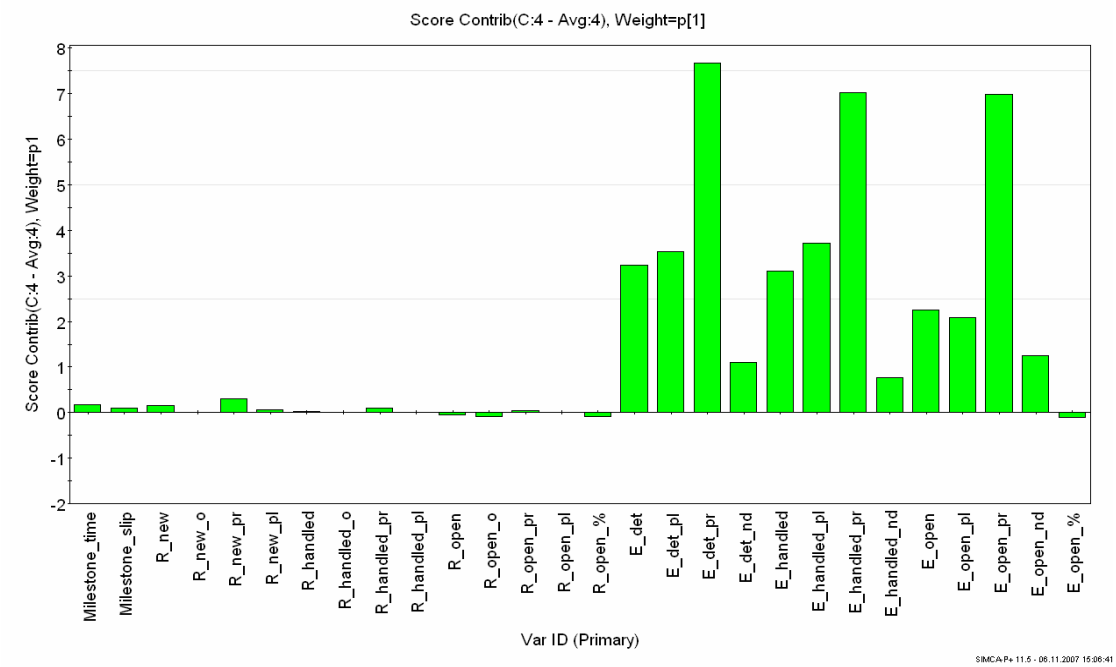


Figure 36: The contribution plot of product C at M1.

product process. In this simulated data, all the errors were handled during the milestone they were detected, and no errors were detected after M4, since it is desirable that there are no severe errors when the product is already on the market. It is noteworthy, that both, in the simulated and in the real product, there is an equal amount of errors, only the handling rate is different.

The outcome of the simulation can be seen in Figure 37 where the orange line represents the real product and the black line the simulated product. The figure suggests that the handling rate can change the product path and make it better follow the average graph.

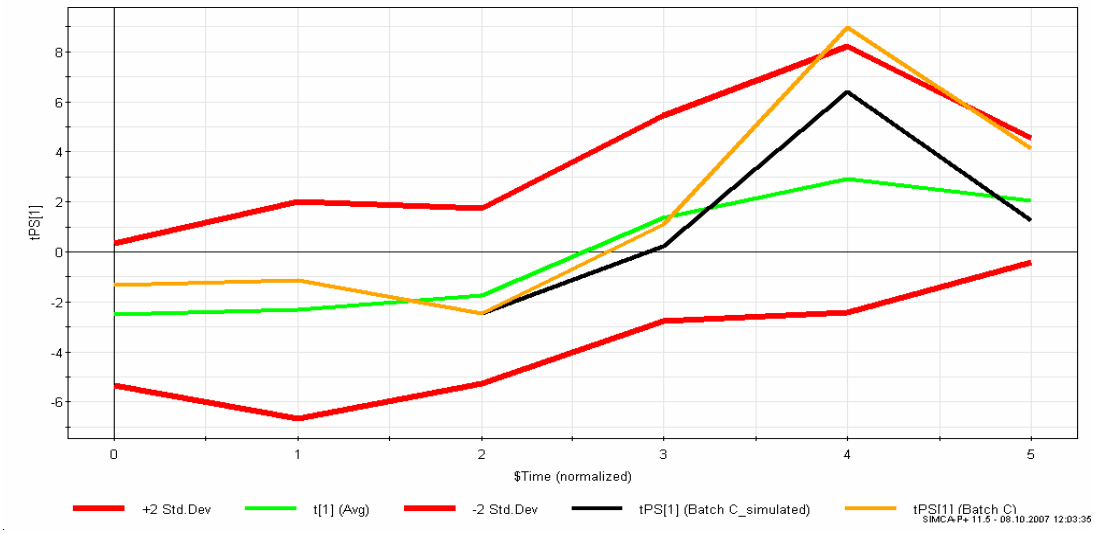


Figure 37: The real product (orange) C and the simulated version of it (black).

## 7.4 Results from interviews

People were selected for the interviews where the batch statistical process control model was presented, if they had a long experience of product programs at Nokia or if they had been closely involved with the product creation processes of Nokia. People belonging to the former group were Matti Alen, Mikko Halttunen, Markku Hiltunen and Eero Juustila and to the latter group Pekka Laaksonen, Pekka Lähteinen and Aarni Soininen.

The old/current product managers generally thought that the selected products A and B, had really been exceptionally dependent on their platform and that the problems of the platform had slowed somewhat the product creation of these two products. In other words, the model was describing their situation. However, it was also stated that such problems were not likely to happen again in the future, since a lesson had definitely been learned: a product can not be that dependent on platform.

Some product managers especially emphasized that the model could serve as a concrete proof for the situation of the product. This means that the product managers have a some kind of general picture in their heads, but the situation would be even easier to describe to others, for example the management, if there was something also in black and white.

The product managers also said that the model could give some additional information about the situation of the products, since people can compare similar products with each other. This means that for example in M3, a product manager can compare his or her own product with a former product that is already on the market and check what were the requirement and error situations then and was the product launched on time and so on.

The process people, on the other hand, were concentrating on a bit more different aspects. It was suggested that the amount of requirement categories to the model should be increased. This means that in addition to the current functional requirements there should also be user requirements in the model from which the functional requirements are derived and then there should be the technology requirements that are derived from the functional requirements (Figure 38).

The importance of efficient requirement managing/handling was also emphasized and the batch model was considered to be a very clear and fast way to see the current

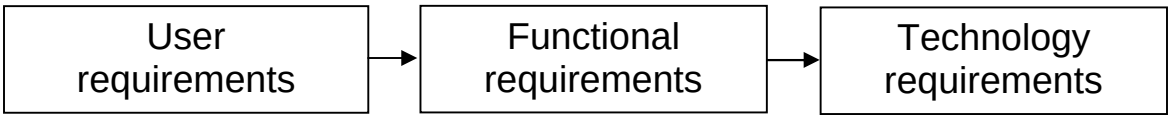


Figure 38: The flow of requirements.

situation. It was also stressed that the error detection should be started as early as possible in the product creation process.

In addition there was a suggestion that the model could be extended in the portfolio level, so that instead of product programs the monitoring could be done in separate portfolios. In practice this would need a quite long history of portfolio management because in order to get a reliable model there should be about twenty rather similar portfolios.

# 8. DISCUSSION

## 8.1 The benefits of the model

The most important benefit of the batch model is its ability to monitor from different angles of the product development process while it is still in progress. This way the problems can be identified and removed before they form an obstacle in the process. The model also shows how the similar kinds of products have evolved and gives some sort of guidelines that way.

The batch model can also serve as a concrete proof of the product development situation, that is, it tells objectively for example if there are alarming amount of requirements open or not. Besides, it can show the overall status of the all products within one portfolio or software platform etc. in only one picture.

The model can also help to spot if there are some systematic problems in the product development process. For example, if the products have always problems with certain errors or if they have always crossed the control limit lines in a certain milestone.

## 8.2 Challenges of the model

The biggest problem of the model is without a doubt the fact that it enables monitoring the product only in 6 time points. The model would be more useful if the monitoring could be done continuously. However, the current batch model can be extended to have an estimate about the next milestone, which basically means that the data will be more accurate as the milestone approaches.

Another problem is that the division of requirements is quite generic at the moment. It is not enough that product managers know the platform to be responsible for the delayed requirements. They know it already. Consequently, the model can be rather used to model the current state of the product process development than as an everyday tool for product managers.

In addition, the model is generated based on the current Nokia products. This does not automatically mean that the current products are made in the best possible way, thus it is not necessarily ideal to build the model on top of them, but rather on top of an ideal product process, where there has clearly been stated what kind of a process

path is allowed, e.g. how big percentage of the requirements have to be defined at each milestone etc.

What is more, the current data set is comprised from quite diverse products, which means that their comparison is not completely justified. However, from the data point of view, only approximately 40 models are launched every year, so even with few categories, it would be demanding to get enough data. This is also because old products can not be used in the modeling since the contents of a mobile phone have changed drastically in just a few years, the data would simply be out of date.

## 8.3 Further prospects

As mentioned in the previous chapter, one of the challenges of this product process development model is its generic nature. Thus it would be beneficial to divide the requirement responsibles and error responsibles into smaller subgroups, into team level. This way it is possible to immediately point out where the bottleneck often is and then ameliorate the situation there. Especially with errors, it could be beneficial to see where errors are most often detected and then by solving the errors earlier, a lot of money could be saved since error correction becomes more and more expensive as the product becomes more mature.

In addition, the requirements could also be divided into subcategories as interviewees suggested. This means that in the beginning are the broader user requirements defined that is what the customer wants to have in his or her phone, and later on these requirements would be divided into functional requirements and finally into technical requirements.

With these two additions one could advance in hierarchy in two dimensions: from big organizations into smaller organizations and from big needs into smaller needs.

The model could be extended to portfolio level, as was also suggested. The key question would be then to think what kind of variables would be most suitable to describe the situation of product portfolio.

*The batch level* modeling was left out of this study is. The batch level modeling would have enabled the prediction of e.g. final product quality, like the field failure rate (FFR), or the product development time. It would be highly beneficial to model

how the amounts of requirements or errors affect the time and quality of the product. This was, however, left out because of time constraints and limited data.

In addition the variables of the batch model could have been weighted in order to notice immediately if there is even slight change to a bad direction. For example if the development time is a key target, the milestone lengths and slips could have been weighted respectively. On the other hand, if the platform dependency is something that should be detected as soon as possible, the weights of platform variables can be altered to bigger ones.

# REFERENCES


1. Albany (1998). University at Albany / SUNY, *A survey of system development process models. Report* CTG.MFA – 003, 13 p., http://demo.ctg-.albany.-edu/publications/reports/survey_of_sysdev/, referred August 27, 2003.

2. Antti H., Bollard M. E., Ebbels T., Keun H., Lindon J. C., Nicholson J. K., Holmes E. (2002), *Batch statistical processing of 1H NMR-derived urinary spectral data*, Journal of Chemometrics, Vol. 16, pp. 461-468.

3. Beck K. (1999), *Embracing Change with Extreme Programming*, Computer, Vol. 32, No. 10, pp. 70-77.

4. Boehm B. W. (1981), *Software engineering economics*, Englewood cliffs, NJ: Prentice-Hall Inc.

5. Börjesson A., Mathiessen L. (2004), *Successful Process Implementation*, IEEE Software, Vol. 2, No. 4, pp. 36-44.

6. Brown M. G., Svenson R. A. (1998), *Measuring R&D Productivity*, Research-Technology Management, Vol. 41, No. 6, pp. 30-35.

7. Chiesa V., Masella C. (1996), *Searching for an effective measure of R&D performance*, Management Decision, Vol. 34., No. 7, pp. 49-57.

8. Cusumano M., Smith S. (1995), *Beyond the Waterfall: Software Development at Microsoft*. MIT Sloan School of Management and International Business Machines, Working Paper #3844-BPS-95, August 16.

9. Damian D., Chisan J. (2006), *En Empirical Study of the Complex Relationships between Requirements Engineering Processes and Other Processes that Lead to Payoffs in Productivity, Quality, and Risk Management*, IEEE Transactions on Software Engineering, Vol. 32, No.7, pp. 433-453.

10.Damian D., Chisan J., Vaidyanathasamy L., Pal Y. (2005), *Requirements Engineering and Downstream Software Development: Findings from a Case Study*, Empirical Software Engineering, Vol. 10, pp. 255-283.

11. Eppinger S. D. (2001), *Innovation at the speed of information*, Harvard Business Review, Vol. 79, No. 1, pp. 149-158.

12. Eriksson L., Antti H., Gottfries J., Holmes E., Johansson E., Lindgren F., Long I., Lundstedt T., Trygg J., Wold S. (2004), *Using chemometrics for navigating in the large data sets of genomics, proteomics, and metabonics (gpm)*, Analytical and Bioanalytical Chemistry, Vol. 380, No. 3, 419-429.

13. Eriksson L., Johansson E., Kettaneh-Wold N., Wold S. (2001), *Multi- and Megavariate Data Analysis*, Umetrics AB, Umeå.

14. Francis P. H. (1992), *Putting Quality into the R&D process*, Research Technology Management, Vol. 35, No. 4, pp. 16-23.

15. Fujii T., Kambayashi Y. (2002), *Strategies to Suppress Productivity Degradation with Unknown Issues under Iterative Development Process*, Proceedings of the First International Symposium on Cyber Worlds.

16. Halman J. I. M., Hofer A. P., van Vuuren W. (2003), *Platform-Driven Development of Product Families: Linking Theory with Practice*, Journal of Product Innovation Management, Vol. 20, pp. 149-162.

17. Hauser J. R. (2000), *Metrics Thermostat*, Journal of Product Innovation Management, Vol. 18, No 3, pp. 134-153.

18. Helo P., Hilmola O-P., Maunuksela A. (2000), *Modelling product development productivity with system dynamics.* The 18th International Conference of the Systems Dynamics Society (Bergen Norway).

19. Hodge M.H. (1963), *Rate your company's research productivity*. Harvard business review, Nov-Dec, pp. 102-122.

20. Kerssens-van Drongelen I., Nixon B., Pearson A. (2000), *Performance measurement in industrial R&D*, International Journal of Management Reviews, Vol. 2, No. 2, pp. 111-143.

21. Ketola P. (2002), *Integration usability with concurrent engineering in mobile phone development*, Dissertation thesis, Department of Computer and Information Sciences, University of Tampere.

22. Kolarik W. J. (1995), *Creating Quality – Concepts, Systems, strategies, and tools*, McGraw-Hill, Inc.

23. Kotonya G., Sommerville I. (1998), *Requirements Engineering – Processes and Techniques*, John Wiley & Sons Ltd, New York.

24. Krishnan V., Gupta S. (2001), *Appropriateness and Impact of Platform-Based Product Development*, Management Science, Vol. 47, No. 1, pp. 52-68.

25. Lintinen M., Siimes T. (2004), *Statistical Multivariate Analysis in the Modeling and Verification of Complex Software Intensive Systems*, Conference on Systems Engineering Research, 15-16 April 2004.

26. McGrath M. E. (1996), *Setting the PACE® in Product Development Revised Edition- A Guide to Product And Cycle-time Excellence®*, Butterworth-Heinemann, Newton.

27. Medhat S. S., Rook J. L. (1997), *Concurrent Engineering – Processes and Techniques for the Agile Manufacturing Enterprise*, 5th International Conference on FACTORY 2000, 2-4 April 1997, Conference Publication No. 435.

28. Meyer M. H. (1997), *Revitalize your product lines through continuous platform renewal*, Research Technology Management, Vol. 40, No. 2, pp. 17-29.

29. Meyer M. H., Lehnerd A. P. (1997), *The power of product platforms: building value and cost leadership*, Free Press, New York.

30. Muffato M. (1999), *Introducing a platform strategy in product development*, International Journal of Production Economics, Vol. 60-61, pp. 145-153.

31. Muffato M., Rowenda M. (2000), *Developing product platforms: analysis of the development process*, Technovation, Vol. 20, pp. 617-630.

32. Nokia (2006), Nokia in 2006: http://www.nokia.com/NOKIA_COM_1/About_Nokia/Sidebars_new_concept/Annual_Accounts_2006/Nokia_in_2006.pdf, accessed 30.10.2007.

33. Olkkonen T. (1993), *Johdatus teollisuuustalouden tutkimustyöhön*. Helsinki, University of Technology, Otaniemi. Industrial Economics and Industrial Psychology, Report No 52

34. Paulk N. C. (2001). *Extreme programming from a CMM perspective*. IEEE Software, Vol. 18, No. 6, pp. 19-26.

35. Perttula A. (2007), *Challenge and Improvements of Verification and Validation Activities in high Volume Electronics Product Development*, Doctoral thesis, Tampere University of Technology.

36. Pesonen L. T. T. (2001), *Implementation of Design to Profit in a Complex and Dynamic Business Context*, Doctoral thesis, Department of Process and Environmental Engineering, University of Oulu.

37. Robertson D., Ulrich K. (1998), *Planning for Product Platforms*, Sloan Management Review, Vol. 39, No. 4,

38. Royce W. W. (1970)., *Managing the development of large software systems: Concepts and techniques.* Proceedings of WESCON. Los Alamitos, CA: IEEE Computer Society Press, Reprinted at the International Conference on Software Engineering '87, Montrey, CA, USA, 30 March- 2 April, pp. 328-338.

39. Sanderson S., Uzumeri M. (1995), *Managing product families: The case of Sony Walkman, Research Polic*y Vol. 24, pp. 761-782.

40. Sheriff A., (1998), *The evolving product platform in FIAT Auto*. In: Presented at the 5th EIASM International Conference on New Product Development, Como, May 25.

41. Simpson T. W., Maier J. R. A., Mistree F. (2001), *Product platform design: method and application*, Research in Engineering Design, Vol. 13, pp. 2-22.

42. Spayd M. K. (2003), *Evolving agile in the enterprise: Implementing XP on a grand scale*. Agile Development Conference 2003. Proceedings June 2003, p. 60-70.

43. Thamhain H. J. (1995), *Applying Stage-Gate Processes in Concurrent Engineering,* IEEE Press, New York.

44. Tipping J. W., Zeffren E. (1995), Assessing *the value of your technology,* Research Technology Management, Vol. 38, No.5, pp. 22-39.

45. Uzumeri M., Sanderson S. (1995), *A framework for model and product family competition*, Research Policy, Vol. 24, pp..583-607.

46. Vilkman H. (2002), *Modular Platform Planning for Mobile Phones*, Master's thesis, Department of Mechanical Engineering, Tampere University of Technology.

47. Winner R.I., Pennell J.P., Bertrand H.E., Slusarzuk, Marko M.G. (1988). *The role of concurrent engineering on weapon systems acquisition.* Institute of Defense Analyses Report R-338. Dec. 1988.

48. Westland J. C. (2002), The cost of errors in software development: evidence from industry, The Journal of Systems and Software, Vol. 62, pp. 1-9.

49. Wheelwright S., Clark K. B. (1992), *Revolutionizing product development: quantum leaps in speed, efficiency, and quality*. Free Press, New York.