# Modeling, Evaluation and Predicting

# of

# IT Human Resources Performance

**Dissertation**

zur Erlangung des akademischen Grades

**Doktoringenieurin (Dr.-Ing.)**

vorgelegt der Fakultät für Informatik

der Otto-von-Guericke-Universität Magdeburg

von M.Sc. Konstantina Richter, geb. Georgieva

geb. am 30.04.1982 in Varna, Bulgarien

Magdeburg, den 30.05.2012

# Contents

# List of Figures

# List of Tables

# 1 Chapter - Introduction

## 1.1 Motivation

In the nowadays era of globalization, based especially on the computer media and applications, organizations of all industrial sectors have to face various problems in order to be successful on the market. Competitors have to respond to the demands for low prices and high quality, along with bright service capabilities and short development life-cycle. It is obvious that these demands are almost impossible to be met and because of this the requirements against the employees in the software development are continuously growing up. The employers are demanding more and more but very often they choose the inappropriate person for a particular job or expect results that are beyond the capabilities of the particular employee.

The subject-matter guru Capers Jones (Jones 2001) characterizes the sad state of software production efforts today and summarizes: ‘’In general, software is a troubled technology plagued by project failures, cost overruns, schedule overruns and poor quality levels. Even companies as Microsoft have trouble meeting published commitments or shipping trouble-free software.’’

So, here raises the question how can we help the software industry? How can we support the software development process? There exist innumerable variety of methods that are meant to be used in the process of development, but the main resource for every company - *the people* seem to be left aside as a point of optimization.

In the nowadays development the importance is concentrated over the hardware and software. Money, time and ideas are invested for new software and hardware achievements, but no one takes care about the third component that is also very important for the successful software engineering – *the people*. The employees are left alone, to manage on their own with the new situation. Methods evaluating the influence of the individuals over the software process do not exist and in this way everything is left to happen by itself.

Because of this the objective of this thesis is to develop a model that is able to evaluate the employees’ performance. This method will assist in the process of personnel acquisition and in this manner will introduce better quality in the software engineering process. The right people chosen in the right manner and also their motivation are the most important software resources, crucial for the achieving of better results.

The work quality in the today software companies is extremely important. It is the basis for everything else and as we have already explained, as the goal is to develop fast and cheap, the people and the way that they work are becoming an inseparable part of the good software development process. Practical applications can be seen in the wide accepted methods for optimization like process maturity (CMMI) and *personal processes* (PSP/TSP). ‘’Adopting PSP and TSP can be a very effective method for accelerating an organization’s progress to higher CMMI maturity levels.’’ This idea, proposed by the SEI shows how important is the process of personnel elaboration and shows also where our model is meant to be applied. Another point that motivated our research and that is also very important for the application of our model is the AGILE development. This new type of software development-organization, extremely dependent from communication and personality types shows once again the importance of the human traits in the software process.

There is one more point that strengthens our motivation: the well-known list with risks from Peter Neumann (Neumann 1985).

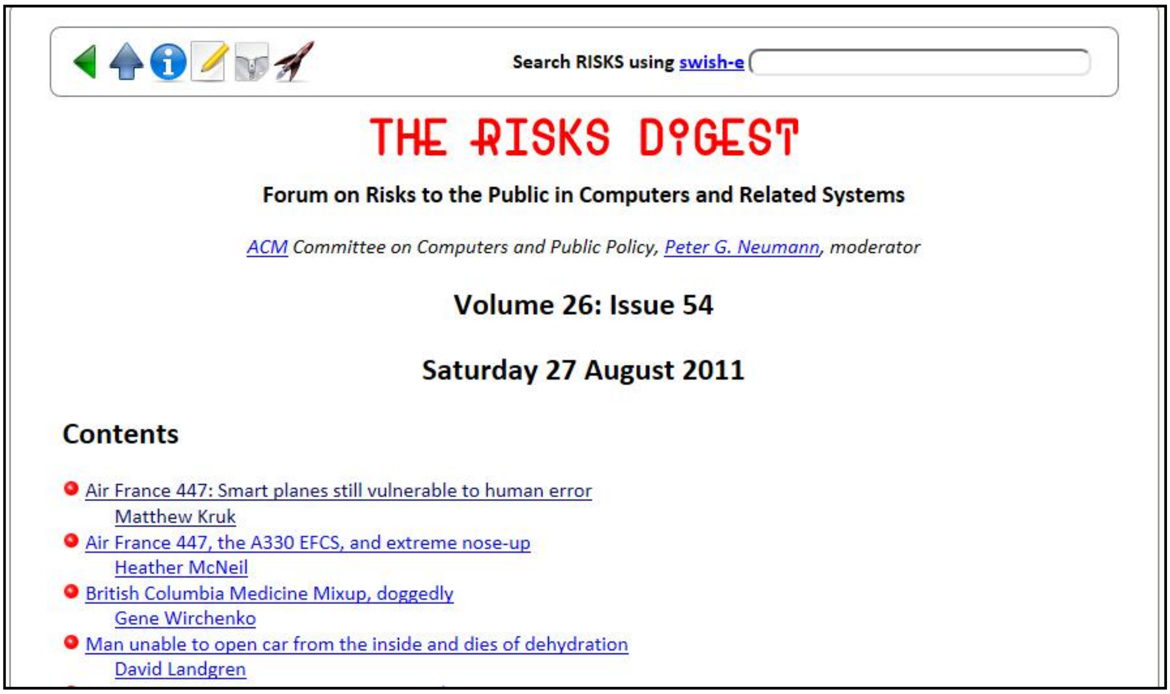



**Figure 1 The Risks Digest**

In the 26-th volume from 27.08.2011 (fig.1) we can see shocking news: ‘’Air France 447: Smart planes still vulnerable to human error - On flight 447, the handoff from computer to pilots proved fatal for the 228 aboard.’’ (Neumann 1985) Exactly this *human mistake* motivates our statement that choosing personnel with a defined psychological profile can be crucial for the performance in a particular software firma and even life-deciding like in this accident.

Led by these ideas our research went through many different stages: from looking for existing similar methods in other fields to adopting engineering solutions in order to find the most *important human characteristics* in the software development and to the end – the development of a method that is able to prognosticate an individual’s performance based on his/her special traits.

In order to fulfill this complex task we had to go through the following steps:

1. Investigation about the existing Software Risk Assessment Methods in order to find out if they cover the Human Factors – Chapter 2.
2. Summarizing all different methods for investigation of the Human slips, mistakes and errors and looking for existing methods that evaluate the human influence in the Software Process – Chapter 2.

3. Investigating the basics of the software engineering in order to find where the personnel take critical part – Chapter 3.
4. Summarizing the basic software team roles and examining their responsibilities – Chapter 3.
5. Adopting the FMEA method for the software engineering needs in order to find the failure modes conducted from the software personnel and in this way the influencing human factors – Chapter 4.
6. Finding a method that could be adopted for the evaluation of the specified in Chapters 2, 3 and 4 human factors. By adoption of the Big Five theory for the software personnel we were able to measure the most important human traits and to observe their influence over the Software Performance – Chapter 5.
7. Evaluating the human traits and choosing of a specific method for estimation of their influence over the IT human resources performance. We used them as input factors for designing of experiment, used to develop a predictive mathematical model for the Human Productiveness – Chapter 6.
8. Validating the gained method for prediction of the IT human performance based on the individuals' characteristics and evaluating its effectiveness and correctness in real conditions – Chapter 7.

## 1.2 Structure of the Thesis

While the introductory chapter of the thesis is concerned with the problem's motivation, the subsequent chapters will focus on the development steps of the proposed method and its validation.

The remainder of the thesis is structured in the following way:

*Chapter 2* examines and investigates the risk management field in the years and after this focuses on the analysis of the risk assessment methods, in order to find out their incompleteness. In the second part of this chapter are discussed the human factors in the software engineering. There is given an overview of different types of characterizations for the human errors, mistakes and failures and the influencing factors are brought to light. This chapter is the basis for the following research as it reveals the problem of ignoring the critical influence of the IT human factors in the software development. It concludes with the observation that there does not exist an adequate method or model that can be used for IT human performance evaluation.

*Chapter 3* is concerned with the software engineering background on which the thesis is build. After explaining the different parts of the software engineering field a deeper look in the software organizations is taken. In this way we were able to find out the most common organizational structure in the software field with its roles and the corresponding responsibilities. The analysis of the IT roles with their competencies and responsibilities is used as basis for specifying the important human factors in the process of software engineering, which are input for the next chapter.

*Chapter 4* examines a well-known method for failure analysis - the FMEA (Failure Mode and Effect Analysis) and adopts it for the need of the software engineering. With the adoption of the method and with the discovered competencies (specified in the previous two chapters) we were able to analyze the roles in the software development process and to find the failure modes by every role and the standing behind specific human factors. The discovered IT human features will take part in the further evaluation process of the human performance.

*Chapter 5* deals with the finding and adopting of a special theory that can evaluate the already found human traits (from chapters 2, 3 and 4) and can also estimate the employee performance in connection with them. These already specified human factors are estimated, using the possibilities of the Big Five Theory. It gives us the opportunity to match the already discovered human factors to the special psychological traits and to visualize the dependence between them and the individuals' productiveness.

*Chapter 6* reflects the development of the specific model for IT human performance evaluation. The discovered (in chapter 5) dependence between the personal factors and the productiveness had to be modeled in an experimental way. For this goal we have chosen a specific experimental design – Design of Experiments as it gives the possibility to find the connection between different factors with a limited number of trials. The result from the chapter is the obtaining of the adequate model that describes the employee performance in a predictive way.

*Chapter 7* is the validation of the developed prognostic models for prediction of the IT humans' performance. There are shown real case studies and a specific web-application, which was developed as implementation of the new model. They all prove once again the accurateness and adequacy of the developed method and show its extreme importance for improving the quality in the software engineering process.

*Conclusion and Future Work* summarizes once again the results and the main contributions of the thesis and gives proposals for further development and application of the model.

# 2 Chapter - Software Risk Management and Human Factors

In the following chapter we are focusing first on the risk management in general and after this on the special risk assessment methods. We are investigating their mechanisms and the data that they are using and in this way we were able to find their incompleteness in the sense that they don't consider the personnel as a crucial part in the risk management process. Based on that observation we are continuing with research about existing methods, taxonomies and types of human factors that play the role of risks in the software development. We end with summarizing the influencing factors for the employee's mistakes and failures and we use this data as foundation for our further research.

## 2.1 Overview over the Development of the Risk Management

Trying to catch the complete history of risk management in the software engineering we have to start from the first attempts made in this field by Nolan (Nolan 1973) (Nolan 1979) and McFarlan (McFarlan 1974), they proposed models for managing the risks in the information systems. In the late 70's Alter and Ginzberg (Alter & Ginzberg 1978) prognosed that risk factor analysis can increase the success rate in the software development. In 1982 Davis (Davis 1982) announces a new method based on requirements determination for selecting the most suitable development approach.
Despite these attempts, risks in their real scope were not addressed until the late 80's, when the pioneer in the software risk management Barry Boehm published his first and most fundamental approach ''A Spiral Model of Software Development and Enhancement'' (Boehm 1988). Later on his work has been complemented by Charette and others (Charette 1989), (Boehm & Ross 1989), (Charette 1990), (Ould 1990), (Boehm 1991). These fundamental works are used later on from the Software Engineering Institute (SEI) (Van Scoy 1992), (Carr et al. 1993), (Higuera et al. 1994), (Higuera & Haimes 1996), (Sisti & Joseph 1994), (Dorofee et al. 1996) for developing a new methodology for risk management based on risk taxonomies.
Other approaches for software risk management are invented from Karolak, Michaels, Pandelios and Hefner (Karolak 1996), (Michaels 1996), (Pandelios, Rumsey & Dorofee 1996), (Hefner 1994). There exist also several risk categories and taxonomies proposed in the fundamental methods of Boehm and SEI. In our paper (Georgieva, Farooq & Dumke 2009 a) we make a summary over existing software development risks and propose new risk taxonomy for the software testing process. Other quantitative approaches appear in the middle of the 90's from Bowers, Fairley and Berny (Bowers 1994), (Fairley 1994), (Berny & Townsend 1993). Kontio proposes a new method for risk management (Kontio 1997), (Kontio 2001) where he proposes risk scenarios that are built over six elements (risk factor, risk event, risk outcome, reaction, risk effect set and utility loss).

In the late 90's and after that several approaches for software risk analysis have been developed separately from the famous ones and they are summarized in our paper (Georgieva, Farooq & Dumke 2009 b), exactly because they are used for risk analysis, which is a part from the risk management we will consider them with special attention and they will be the milestone for our scientific motivation.

A small number of industrial reports have been published, so we will give just few examples: (Boehm 1991), (Chittister, Kirkpatrick & Van Scoy 1992), (Eslinger et al. 1993), (Meyers &

Trbovich 1993), (Morin 1993), (Fairley 1994), (Gemmer & Koch 1994), (Hefner 1994), (Williamson 1994), (Conrow & Shishido 1997).

Many different risk assessment frameworks were proposed in the years. For example McComb and Smith' framework that identifies system failure factors, covering 15 key risk areas - from project planning and execution to technical and human factors (McComb & Smith 1991). Barki et al., based on a literature survey over 120 projects composed a list of 35 features, connected with the software development risk (Barki, Rivard & Talbot 1993).

Thomsett (Thomsett 1992) invented a risk assessment questionnaire model and proposes a new project management paradigm that recognizes people-oriented values as very important in the traditional organization structure and with this he is one of the first that puts accent on the people in the process of risk management.

SEI risk taxonomy, already mentioned before is an important contribution in the field of risk management because it gives a very comprehensive questionnaire and software risk evaluation method (Carr et al. 1993), (Sisti & Joseph 1994). Another risk assessment framework is proposed by Lyytinen (Lyytinen, Mathiassen & Ropponen 1996) and later on is enlarged from Keil (Keil et al. 1998).

Applegate (Applegate, McFarlan & McKenney 1996) publishes a book about Information Systems management, where the project risk assessment questionnaire is the tool to evaluate the risk-degree in the different IT applications. Another method was developed by Moynihan, who collected a list of risks and planned their mitigation after interviewing particular project managers (Moynihan 1997), (Moynihan 2002). Project failure because of unmanaged risk is widely recognized theme in the project management community. The general process and principles of project risk management are applicable to all kinds of software projects. There is quite extensive literature on generic project risk management and we name only the most comprehensive works such as (Wideman 1998), (Chong & Brown 2000) (Pritchard 2001), (Chapman & Ward 2002), (Chapman & Ward 2003), (Kendrick 2003), (Mulcahy 2003) and (Smith & Merritt 2002). The last trends are to extend the risk management over safety, environmental and business risk (Waring & A.I. 1998), (Cooper et al. 2004) or to addresses the so called 'positive risk' (Hillson 2004).

If we have to make an observation about the evolution of the software risk lists in the last two decades than we have to start with McComb's 50 issues (McComb & Smith 1991), (Barki, Rivard & Talbot 1993). After that Thomsett created a bigger questionnaire (Thomsett 1992) and the most famous questionnaire for software project risk originates from the SEI (Carr et al. 1993). McConell creates another risk identification questionnaire but focused on the software code and schedule (McConnell 1993), (McConnell 1996). In the well-known book of Capers Jones can be found a list of 60 software project risks (Jones 1994). Lyytinen creates also a questionnaire covering the main software development risks (Lyytinen 2000). Cockburn has summarized some of the current knowledge on effective risk management strategies into reusable risk resolution patterns (Cockburn 1997).

The Software Engineering Institute stresses their research on the importance of the teamwork in the risk management and as a result they have united their ideas into a Team Risk Management method (Higuera et al. 1994). Another work in this direction is from Kontio, who examines the effectiveness of the group work in his method Riskit.

In the dynamic world that we are living in, the risk management is recognized to be a major part of the successful software engineering and because of this it is covered by all the 'bibles' of software engineering and project management such as (CMMI 2002), (Thayer & Dorfman 2002) (Pressman 2004), (Sommerville 2004), (McConnell 2004), (Abran & Moore 2004), (PMBOK 2004).

These important milestones in the software risk management give us a solid basis to motivate our research work. We have seen the lack of methods for evaluation of the human productivity in the software development process and in the same time we were able to recognize the major importance of the human factors as a crucial risk element. So for us the idea to develop a method for evaluation of the human performance was a logical conclusion.

## 2.2 The Incompleteness of the Risk Assessment Methods

Risk assessment methods are one of the most important elements in the process of risk management. These methods consider numerous aspects while assessing and estimating the risks. Since software development is a human intensive activity, diverse factors related to human behavior also play a key role in this situation. Software risk assessment methods should take into account all these factors in combination to each other. Because of this here we will have a short view over the current applied risk assessment methods and their consideration of human factors.

Observing the principles of risk management given by the International Organization for Standardization, described in ISO/FDIS 31000 (ISO 2009) it is clear to see the following statement:

> '*Risk management should take into account human factors. The organization's risk management should recognize the capabilities, perceptions and intentions of external and internal people that may facilitate or hinder attainment of the organization's objectives*'.

This statement gives a strong motivation to our thesis that the human factors are in the center of the risk management process and that they should be a part of the risk assessment methods. Other evidences emphasizing the role of human factors in software engineering and software development process include the People Capability Maturity Model and the pair programming development technique.

Based on the Boehm's classification of risk management we will focus on methods for risk analysis and the lack of consideration of the human factors in them. The methods for risk assessment are very important in the process of risk management because they give the possibility to predict the success of a particular project. Realizing their crucial role in the process of risk management we have to realize also that the main actors in every process are the humans, and their actions give rise to different issues or problem situations. We can have a look over a simple example: in the medicine, the safety of different machines is maintained by people. So it is clear to see how important the people in this case are. Any mistake can lead to a death of a patient. It is the same in the software development process, any risk brought by a human can be crucial for the whole system.

The risk assessment since 1995 has been shortly summarized in the following part of this chapter in order to see the mechanism of work. The risk assessment methods are very different by their nature: they explore different structures in the software development process, use different techniques, and are applied over different phases in the development process. So, we are able to see a great variety of techniques. The methods will particularly be investigated for their consideration of human factors while assessing and estimating risks. Our goal is to put a stress on the importance of the humans in the development process as the people stay at the basic level and they should not be underestimated. It is not possible to achieve a complete risk assessment, or risk management over a system if we do not include in it also the human factors.

There exist different types of human factors studies: human error analysis, human factors engineering and human reliability analysis (Baybutt 1996). The errors that people commit can be seen in different perspectives, for example in the process of work of: people with other people, people with equipment/with procedures, tasks and others. A basic classification of the human errors (Baybutt 1996) distinguishes between: slips, mistakes, violations, socio-technical and coming from the management. We will describe all different types of problems caused by the employees later on in this chapter.

The following methods for risk assessment are grouped according to the base technique that they use in the process of assessing the risks. Every method is described shortly and is analyzed for all types of factors that it considers with a special focus on existence of human factors among them.

### 2.2.1 Neural Networks Based Risk Analysis Methods

Artificial Neural Networks (ANN or just Neural Networks (NN)) are modeled after the biological neurons in brain structures. The individual neuron models may be combined into various networks made up of many individual nodes, each with its own set of variables. These networks have an input layer, an output layer, and one or more hidden layers. The hidden layers provide connectivity between the inputs and outputs. The network may also have feedback, which will take result variables and use them as input to prior processing nodes. With the help of NN it is possible to be modeled different possible directions in the process of software development and in this way to find the potential risks.

*Using Influence Diagrams for Software Risk Analysis* (Chee, Vij & Ramamoorthy 1995)

**Input:** software metrics data collected at various stages of software development
**Technology:** influence diagrams, kinds of NN, used for probabilistic and decision analysis models
**How it works:** The method uses the conditional independence implied in the influence diagrams in order to determine the information needed for solving of a problem. Influence diagrams are used to provide quantitative advice for software risk management, improving upon traditional ad-hoc software management techniques.

*An Enhanced Neural Network Technique for Software Risk Analysis* (Neumann 2002)

**Input:** software metric data
**Technology:** principal component analysis and artificial neural networks (PCA-ANN). Uses pattern recognition, multivariate statistics and NN.

**How it works:** This is a technique for risk categorization in which principal component analysis is used for normalizing and orthogonalizing the input data. A neural network is used for risk determination/classification. The special feature in the approach, the so called cross-normalization is used to discriminate data sets, containing disproportionately large numbers of high-risk software modules.

*A Neural Networks approach for Software Risk analysis* (Yong et al. 2006)

**Input:** software risk factors, obtained through interviews/questionnaires
**Technology:** combination of principal component analysis, genetic algorithms and neural networks
**How it works:** Based on the SEI and interviews with professionals in the field, is created taxonomy and factors for software risk. This data after processing is used as an input for the NN analysis. The method is divided in the following steps: 1) predict the risks with standard NN; 2) predict with the combination of NN and PCA; 3) predict with the combination of GA and NN; and 4) combine the three steps and make an overall prediction.

*Analyzing Software System Quality Risk Using Bayesian Belief Network* (Young et al. 2007)

**Input:** project risk factors selected through a Delphi method based on historical project data
**Technology:** Bayesian Belief Network, Delphi method
**How it works:** The method is based on BBN and predicts and analyzes the changing risks of software development based on facts such as project characteristics and two-side (contractors and clients) cooperation capability at the beginning of the project. BBN are used for the analysis of uncertain consequences or risks and Delphi method is used for the network structure needed for the BBN. The method is used to evaluate the software development risks in organizations.

In the system for risk assessment, proposed in the method are considered problems connected with lack of experience among the employees. Anyway we cannot say that the method considers all different human factors, because of the complex nature of the human being.

### 2.2.2 Qualitative Based Risk Analysis Methods

Qualitative methods are methods that take into consideration different qualities. They collect information with the help of different questionnaires. In this way they analyze not numerical but qualitative data and after this based on it give the possibility for risk analysis.

*SRE from the SEI Risk Management Paradigm* (Williams, Pandelios & Behrens 1999)

**Input:** software risk information, obtained through interviews/questionnaires
**Technology:** questionnaires
**How it works:** The SRE addresses the identification, analysis, planning, and communication elements of the SEI Risk Paradigm. The method implies the following:
• trains teams to conduct systematic risk identification, analysis, and mitigation planning
• focuses upon risks that can affect the delivery and quality of software and system products
• provides project manager and personnel with multiple perspectives on identified risks
• creates foundation for continuous and team (customer/supplier) risk management

### 2.2.3 Software Metrics Based Risk Analysis Methods

Software metric is a measure of some software property and it is important to know that the metrics give quantitative information about different characteristics of the software, which could be used for risk analysis.

*Software Risk Assessment and Estimation Model* (Gupta & Sadiq 2008)

**Input:** Measurement error, Model error, Assumption error in function point estimation
**Technology:** risk exposure and Mission Critical Requirements Stability Risk Metrics
**How it works:** The risk is estimated using risk exposure and software metrics of risk management, which are used when there are changes in requirements. Initially the model estimates the sources of uncertainty using Measurement error, Model error and Assumption error.

*A Risk Assessment Model for Software Prototyping Projects* (Nogueira, Luqi & Bhattacharya 2000)

**Input:** requirement, personnel and complexity metrics
**Technology:** different software metrics
**How it works:** The method introduces metrics and a model that can be integrated with prototyping development processes. It claims to address to some extent the issue of human dependency in risk assessment but it is not clear how exactly, because there are no mentioned metrics for that.

*Source-Based Software Risk Assessment* (Deursen & Kuipers 2003)

**Input:** source code information
**Technology:** code metrics, questionnaires
**How it works:** The method focuses on "primary and secondary facts''. Primary facts are obtained through automatically analyzing the source code of a system with code metrics, and secondary facts are obtained from people through different questionnaires, who are working with or on the system. The both type of facts are of different type information, so there is needed a bridging between them and after this the obtained information is used to advise a minimizing of the potential risk.

### 2.2.4 Early Risk Estimation Based Risk Analysis Methods

Analyzes in the early stages of the software development is one of the desired perspectives in the process of risk estimation and mitigation. It is much cheaper if we can encounter and overcome the problems in the early stages as if we do this at a late stage of the software development process.

*A Methodology for Architecture-Level Reliability Risk Analysis* (Yacoub & Ammar 2002)

**Input:** severity of complexity and coupling metrics derived from software architecture
**Technology:** dynamic metrics, architecture elements
**How it works:** That is a heuristic risk assessment methodology for reliability risk assessment, based on dynamic complexity and dynamic coupling metrics that are used to define

complexity factors for the architecture elements. Severity analysis is executed with Failure Mode and Effect Analysis, applied over the architectural models. A combination between severity and complexity factors is used in order to identify the heuristic risk factors for the architecture components and connectors.

*Software Risk in Early Design Method* (Vucovich et al. 2007)

**Input:** software functionality, Historical Function-Failures, Historical Failure Severities
**Technology:** Function-Failure Design Method
**How it works:** This method identifies and analyzes the risk presented by potential software failures. With the Software Function-Failure Design Method it is demonstrated the corresponding Risk in Early Design method to the software domain, to provide a software risk assessment based on functionality, which is often the only available information in the early stages of design. RED allows the early assessment of risk, which can guide more-detailed risk assessment, provide a test-case development guide, and help in deciding on whether a software product has been tested enough.

## 2.3 Summary over the Risk Management and Motivation of our Further Research

Let us summarize the ***risk management*** methods as ***Gallery of Software Risks:***

*Crisis management*: Nolan (Nolan 1973) (Nolan 1979) and McFarlan (McFarlan 1974) proposed models and project portfolio for managing the crisis in the information technology and the risks in the information systems. Alter and Ginzberg (Alter & Ginzberg 1978) proposed that risk factor analysis can increase the success rate in the software development. Davis (Davis 1982) creates a new method based on requirements determination for selecting the most suitable development approach.

*Risk management*: The pioneer in the software risk management Barry Boehm published his first and most fundamental approach ''A Spiral Model of Software Development and Enhancement'' (Boehm 1988) about risk management. Later on his work has been complemented by Charette and others (Charette 1989), (Boehm & Ross 1989), (Charette 1990), (Ould 1990), (Boehm 1991).

*Risks taxonomies*: Several risk categories or taxonomies proposed in the fundamental methods of Boehm and SEI and few others. In the paper (Georgieva, Farooq & Dumke 2009 a) was made a summary over existing software development risks and propose new risk taxonomy for the software testing process. Other quantitative approaches are from Bowers, Fairley and Berny (Bowers 1994), (Fairley 1994), (Berny & Townsend 1993).

*Risk scenarios*: Kontio proposes a new method for risk management (Kontio 1997), (Kontio 2001) where he proposes risk scenarios that are built over six elements (risk factor, risk event, risk outcome, reaction, risk effect set and utility loss).

*Risks analysis:* In the late 90's and after that several approaches for software risk analysis have been developed separately from the famous ones, no matter that they are used for risk analysis they are very specific and cannot be taken as general big methods for risk management.

*Risks experiences*: Risk experience as industrial reports have been published from (Boehm 1991), (Chittister, Kirkpatrick & Van Scoy 1992), (Eslinger et al. 1993), (Meyers & Trbovich 1993), (Morin 1993), (Fairley 1994), (Gemmer & Koch 1994), (Hefner 1994), (Williamson 1994), (Conrow & Shishido 1997).

*Risk frameworks*: Many different risk assessment frameworks were proposed such as the framework of McComb and Smith to identify system failure factors, which includes 15 key risk areas distributed between project planning and execution in one dimension and technical and human factors in the other one (McComb & Smith 1991).

*Risk-based features*: Barki et al., based on a literature survey of 120 projects compiled a list of 35 features that are connected with the software development risk (Barki, Rivard & Talbot 1993).

*Risks assessment*: Thomsett (Thomsett 1992) develops his risk assessment questionnaire model, where all the questions are divided into three areas and each question has a specific value and is later used in forming the final score of risk. He proposes a new project management paradigm that recognizes people-oriented values as very important in the traditional organization structure and with this he is one of the first that puts accent on the people in the process of risk management.

*Risks evaluation*: SEI risk taxonomy, already mentioned before is an important contribution in the field of risk management because it gives a very comprehensive questionnaire and software risk evaluation method (Carr et al. 1993), (Sisti & Joseph 1994).

*Performance-oriented risk management*: Risk assessment framework developed by Lyytinen et al. (Lyytinen, Mathiassen & Ropponen 1996) and later on supplemented from Keil (Keil et al. 1998) presents three-level-structure of management, project and system environment, that gives a performance based on how actors, structure and technology are assembled.

*Risks degrees and experience*: Applegate (Applegate, McFarlan & McKenney 1996) publishes a book about IS (Information Systems) management, where the project risk assessment questionnaire is the tool to evaluate the risk-degree in the different IT applications. Further approaches have been developed by Moynihan, who collected a list of risks and planned their mitigation after interviewing experienced project managers (Moynihan 1997), (Moynihan 2002).

*Risks assessment questionnaires*: Questionnaires-Based frameworks are developed, for example the One-minute Risk Assessment Tool from Tiwana and Keil (Tiwana & Keil 2004-2005). A comparison of selected risk management approaches can be found in (Lyytinen, Mathiassen & Ropponen 1998). Questionnaires and risk lists as a form of risk identification appear from the very beginning and are still the most relevant and used techniques. The first lists comprised less than 50 issues (McComb & Smith 1991), (Barki, Rivard & Talbot 1993).

*Risks factors*: A list of 60 software project risk factors can be found in (Jones 1994), where each factor is analyzed for its frequency, impact, root causes, mitigation strategies and others.

*Risks management teamwork*: The Software Engineering Institute stresses their research on the importance of the teamwork in risk management as Team Risk Management method (Higuera et al. 1994). The effectiveness of the group work (including the brainstorming) has been detailed investigated by Kontio. He developed the Riskit method using communicative and easily distinguishable elements of risk scenarios, which were visualized in a risk analysis diagram (Kontio 1997), (Kontio 2001).

*Quantitative vs. qualitative risks analysis*: Distinguishing between qualitative and quantitative methods, the qualitative techniques estimate the risk in terms of likelihood and impact and apply ordinal scales and risk matrices as well as some means of weighting and averaging of the obtained score (Charette 1990) (Sisti & Joseph 1994) and quantitative risk analysis calculates the risk based on the theories of the probability calculus such as Monte Carlo analysis or Bayesian Belief Networks (Grey 1995) (Vose 2008) (Schuyler 2001). The well-known in the engineering field Failure Mode and Effect Analysis method (FMEA) was applied to the analysis of project risk in (Deept & Ramanamurthy 2004).

*Generic project risk management*: The general process and principles of project risk management are applicable to all kinds of software projects. Examples of generic project risk management are described in (Wideman 1998) (Chong & Brown 2000) (Pritchard 2001), (Chapman & Ward 2002) (Chapman & Ward 2003) (Kendrick 2003) (Mulcahy 2003) and (Smith & Merritt 2002).

*Project risk management*: Well-known risk management solutions for software projects are created from Boehm, Karolak and Hall (Boehm 1991), (Karolak 1996), (Hall 1998). On later stage Boehm (Boehm et al. 2003) proposes a risk approach of COTS-intensive projects.

*Risks perception*: Adams gives very important observations on the everyday risk perception and management in (Adams 1995). A practitioner's view on project risk management can be found in (Conrow 2003). Several works are admitted to be actually used and accepted in the software development industry (Ropponen & Lyytinen 2000) (Moynihan 2002).

*Business risks*: Risk management over safety, environmental or business risks are described in (Waring & A.I. 1998) and (Cooper et al. 2004) or to addresses the issue of 'positive risk' of a business opportunity (Hillson 2004). Case studies of business risk management are described in (Schmietendorf 2009).

*Risks management strategies*: Cockburn has summarized some of the current knowledge on effective risk management strategies into reusable risk resolution patterns (Cockburn 1997).

*Risks management database*: Kontio presented a detailed design of a risk management database (Kontio & Basili 1996) (Kontio 2001) but its scope is limited to capturing the information on risk in actual projects and lacks the capabilities to develop generalized knowledge.

*Risk management in software engineering*: Risk management is a main essential part of the management of a successful software project and because of this it is covered by all the 'bibles' of software engineering and project management such as (Chrissis, Konrad & Shrum 2003) (Thayer & Dorfman 2002) (Pressman 2004) (Sommerville 2008) (McConnell 2004) (Abran & Moore 2004) (PMBOK 2004).

*Risks management standards:* The area of risk management is intensely standardized and the most widely recognized risk management standard is ISO 14971 (14971 2001) complemented by IEC 62304 (62304 2004). Although that ISO 14971 covers the risk of medical devices, it is generally accepted as a mature standard on general-purpose risk management. Based on ISO 14971, Standards Australia has proposed a new extended standard AS/NZS 4360 (4360 2004), which is expected to replace the ISO 14971. ISO has also published a risk management standard ISO 16085 dedicated to software engineering (16085 2006), which is based on the earlier work from IEEE - the IEEE 1540 (1540 2001).

*Risks and human factors*: There exist different types of human factors studies: human error analysis, human factors engineering and human reliability analysis (Baybutt 1996). Because of this a basic classification of the human errors (Baybutt 1996) can look in the following way: slips, mistakes, violations, socio-technical and coming from the management.

Observing the described risk assessment methods we can make the following statement: all of them take as input different type of data, that could be generalised like: architecture, design and code metrics data as visualized in Figure 2. Only few of these methods - (Young et al. 2007), (Nogueira, Luqi & Bhattacharya 2000) consider some types of human factors. Although that this attempt does not seem to be comprehensive, it is a good example which gives as much importance to human factors as to the others in the process of assessing software risks.

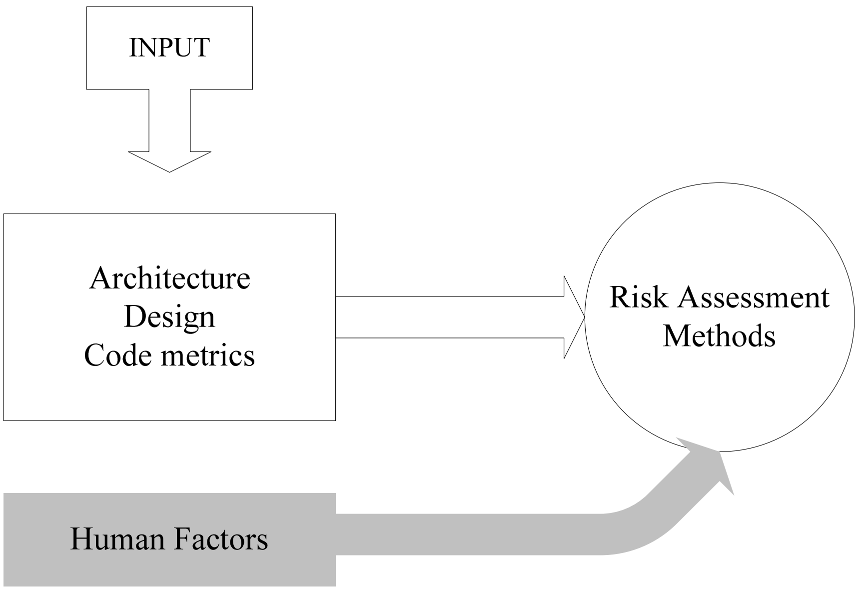



**Figure 2 Input for the Risk Assessment Methods**

As it can be seen in Figure 3 the risk sources in the software production process are: people *P*, development process *D*, software *S* and hardware resources *H*. These four elements give us the complete software development or software production process *SPP* and software system *SS* (as *IT* area), which should be analyzed in its full complexity in order to achieve an adequate risk management process *RM* including the risk assessment *RA* and the risk controlling *RC*.

This can be expressed with the following equations according to (Boehm 1991) and figure 3:

$$IT = \{ SPP , SS \} \qquad (2.1)$$

$$SPP_{riskSources} = \{P_{dev}, D_{dev}, S_{dev}, H_{dev}\},$$

$$SS_{riskSources} = \{P_{sys}, S_{sys}, H_{sys}\}$$

Furthermore, the risk assessment could be considered for both – software development or production and software system as

$$RA_{SPP}: personnel_{dev} \times development_{dev} \times software_{dev} \times hardware_{dev} \rightarrow riskAssessment_{dev} \qquad (2.2)$$

$$RA_{SS}: personnel_{sys} \times software_{sys} \times hardware_{sys} \rightarrow riskAssessment_{sys}$$

And finally, the general components of the risk management as risk assessments and risk controlling as $RM = \{RA, RC\}$ are

$$RA = \{ riskIdentification, riskAnalysis, riskPrioritization \}, \qquad (2.3)$$

$$RC = \{ riskMgmtPlanning, riskResolution, riskMonitoring \}.$$

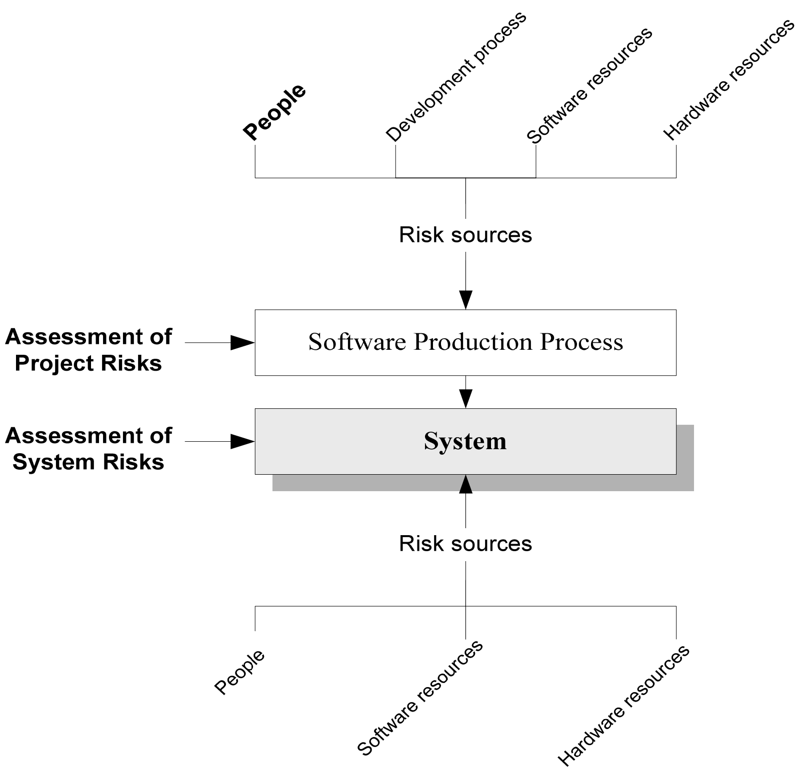



**Figure 3 Risk in the different stages of the development process**

Taking into consideration the information obtained from the analyzed risk assessment methods, which is that they do not consider the people like a major source of risk and analyzing the software system in its complete form and knowing how crucial can be the role of the human being in every activity (Georgieva, 2009 c), we can conclude that there exist an incompleteness of the existing methods for risk assessment and new methods should be developed which cover the human factors.

## 2.4 Human Factors in the Software Engineering

The humanity is what makes the world move forward in a technical, experimental and achieving way. Human skills, ideas and imagination are the inspirations for all surrounding inventions and technologies, cultural, tradition and intellectual progress. Humans develop the technology to a newer level, always higher, always faster and hopefully always better. The trace of human touch and sense is in every emerging technology, theory, business solution and machine and of course when there is a human act – there might be a human error too.

In order to understand the complexity of the human being we will start with a small example from our biological nature. Let us observe the human retina (figure 4). This transparent, paper-thin layer of nerve tissue on which is projected an image of the world, that is less than 1 cm square and a ½ mm thick has about 100 million neurons. The retina processes about ten one-million-point images per second. If we want to simulate this activity with a computer, than it will take him 100 MIPS to do a million detections, and 1,000 MIPS to repeat them ten times per second in order to match the retina. (Moravec 1997)

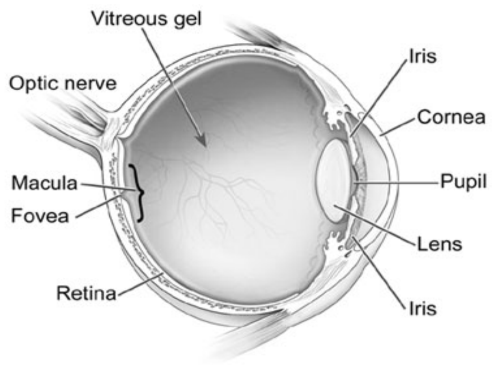



**Figure 4 The human eye, (Human Eye 2011)**

Having this information in mind let us see what is happening in our brain (figure 5). The 1,500 cubic centimeter human brain is about 100,000 times larger than the retina; this means that matching the brain activity will take about 100 million MIPS (million instructions per second) of computer power. (Moravec 1997)

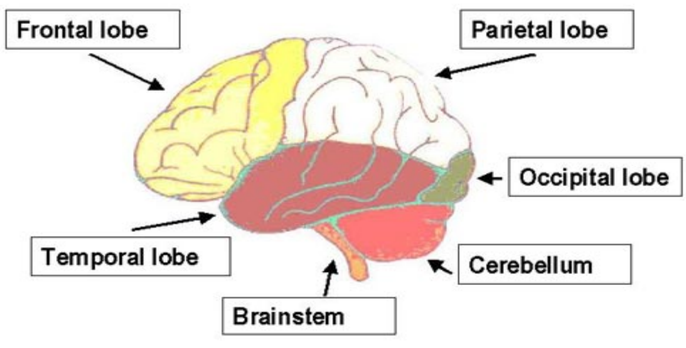



**Figure 5 The human brain, (Human Brain 2011)**

This small observation shows the complexity of the human brain, that we have to take into consideration when speaking about human factors. Here we are just observing the technical parameters of the brain, but when we take also the influencing factors like health, emotions, motivations, ambitions and qualification the overall picture becomes much more complex. This is what motivated us to analyze the connection between the personal characteristics and the human productiveness in the software development process.

### 2.4.1 Human Errors, Mistakes and Failures

Human errors examples might be found everywhere: small quarrels with relatives affected by a complicated character; design problems in a usability form; machine construction and usage; people to people and human to machine interaction. Consequences are also numerous from small frowns and bad attitude to catastrophic life threatening events.

Human error is the difference that occurs from what a human is supposed to make (planed, proposed, intended) and what the result (or lack of it) is. In some cases the difference is so unnoticeable that it stays hidden, sometimes it is discovered and mitigated or remains hidden bringing along unpredictable results when emerging. The factors affecting the result and production of an error are also classified of a human kind. In the following part we have summarized the leading classifications about human factors.

The pioneer in the field of human factors is Rasmussen, he publishes his classification in 1982 and distinguishes between three types of problems that could be divided into: skill-based, rule-based and knowledge-based level (Rasmussen 1982). Skill-based performance is explained with automatic, unconscious and parallel actions. Rule-based is associated with recognizing situations and following associated procedures. Finally, “knowledge-based” refers to conscious problem solving. Rasmussen also proposes a list of factors that influence the human behavior and actions: social and management climate, type of the overworked information, emotional condition, physiological stressors and physical workload. He pioneered a multi-facet taxonomy for the description and analysis of events involving human malfunction. In this taxonomy, he defines the causes of human malfunctions as: “external” (distraction, etc.), “excessive task demand” (force, time, knowledge, etc.), “operator incapacitated” (sickness, etc.), and “intrinsic human variability”. As we will see in the next paragraphs his ideas are completely adopted and slightly modified and extended by Reason and Shappell.

Reason (Reason 1990) has defined the human error as a planned sequence of mental or physical activities to achieve its intended outcome. He distinguishes between *mistakes and slips* and in his view slips are actions that proceed as planned but end with undesired actions and mistakes are desired actions, which go as they are supposed to, but are not fulfilling the planned goal, so they are classified as planning failures or latent failures. Latent failures unlike their active counterparts may remain unnoticed for a long period before emerging in an unsuspecting situation.

From Reason (Reason 1990) descriptions’ of latent and active failures, Shappell (Shappell 2000) distinguishes four levels of failures: *unsafe acts, predictions for unsafe acts, unsafe supervision and organizational influences.* Although that the ‘The “Swiss cheese” model of accident causation’ of Shappell is meant to be used for the aviation it could be applied in the field of software engineering with great success. Anyway almost all of the definitions and research work about human factors is not originating from the software field because the software industry has emerged in the last decades and it was not as popular as the other already developed industries.

Reason separates two types of *unsafe acts: errors and violations*. Errors are described as mental or physical actions of an individual that do not accomplish desired outcomes. And violations are specified as determination not to obey the rules and recommendations that impose safety.

To determine more correctly a specific failure investigation these two categories are expanded as shown here (Figure 6) into three types of errors (skill-based, decision, and perceptual) and two violation forms.

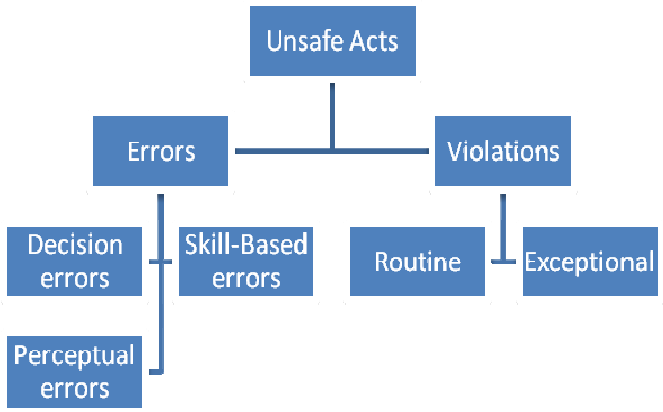



**Figure 6 Unsafe Acts categories (Shappell 2000)**

Examples of these errors and violations we can see in the next Table 1.

**Table 1 Selected examples of Unsafe Acts (Shappell 2000)**

| **Errors** | **Violations** |
|---|---|
| <u>Skill Based Errors</u><br>• Failed to prioritize attention<br>• Omitted step in procedure<br>• Omitted checklist item<br>• Poor technique | Failed to adhere brief<br>Violated training rules<br>Not currently qualified for work |
| <u>Decision Errors</u><br>• Improper procedure<br>• Misdiagnosed emergency<br>• Wrong response to emergency<br>• Exceeded ability<br>• Poor decision | |
| <u>Perceptual Errors</u><br>• Visual illusion<br>• Disorientation | |

We see that all these different errors are based on the individual's skills, decision or knowledge in the special moment, so no matter why these errors occur they are based on the individual perception of the world. This will give us motivation to develop our performance prediction model based exactly on these individual features that make from people with the same knowledge and experience absolutely different employees from the point of view of their performance.

Technical failures (Shappell 2000) also specified as *skill based* errors are based on the individual experience and education. *Decision errors* describe intentional behavior that ends with inappropriate or inadequate action for the situation. *Knowledge based* errors (Perceptual errors) occur when one's perception of the surrounding is different from the reality.
Rasmussen defines also the so called *ruled based* mistakes (Rasmussen 1982) or procedural errors (Orasanu 1993), they occur when a structured task is faced but the wrong procedures are performed.

*Violations* - are produced during intentional disregard of laws and orders. We can have routine and exceptional violations – that occur as a rare withdraws from standard regulations, not demonstrating an individual typical behavior (Shappell 2000).
Observing all these different unsafe acts it is important to understand why they happen and Shappell gives the explanation with different *preconditions*. They can be for example substandard conditions which represent the different mental and physiological state that the people can be in and the resulting from that behavior. There is one more level of failures: *unsafe supervision and organizational influences*, it is extremely important to understand that although the people and their mental state and cooperation are very important, it is also important the way that the company or team is leaded and what kind of atmosphere we have during the working process, we can see an example in (Georgieva et al, 2010 a) (Georgieva, 2009 d).

### 2.4.2 Influencing Factors

We cannot describe the human factors in the software process only as *errors, mistakes and failures* but we have to describe also the many different factors that influence the people in their everyday work and that lead them to successful or not fulfillment of their work. The problems that the people cause are only one facet of the problem that we want to solve. We look actually for the special human features that lead to a bigger or smaller number of problems. Because of this now we will have a look over the rest of the human factors. We have listed some of these factors stated by Shappell, Reason and Rasmussen in the previous section so now we will continue with the following authors:

Fisher (Fisher 2001) tries to summarize what are the important points when we want to create a successful user software system. He identified the following necessary human and technical skills: graphic design, communication, organization of information, illustration, interface design and usability testing.

Wang (Wang 2005) proposes taxonomy of human factors in software engineering and builds a behavioral model of human errors, which is expressed in an evaluation of the performed task. This model concentrates on the conducted by humans actions in the process of performing a certain task. In 2008 (Wang 2008) Wang broads his taxonomy and categorizes the personality traits into eight groups. These can be seen in the following Table 2.

**Table 2 Taxonomy of personal traits and attributes (Wang 2008)**

| Emotion & Motivation | Attitude | Cognitive Ability | Interpersonal Ability |
|---|---|---|---|
| Comfort/fear | Proud of job | Knowledge | Pleasant |
| Joy/sadness | Responsible | Skills | Tolerant |
| Pleasure/anger | Disciplined | Experience | Tactful |
| Love/hate | Thorough | Instructiveness | Helpful |
| Ambition | Careful | Learning ability | Scope of contact |
| Impulsiveness | Assertive | Expressiveness | Variety of contact |
| Trying in uncertainity | Energetic | Knowledge transferability | Consultative |
| Following rules | Enthusiastic | Reaction to events | Responsible |
| Self-expectation | Tolerant | Efficiency | Respectful |
| | Tactful | Attention | Trustworthy |
| | Confident | Abstraction | Sympathetic |
| | Individual | Searching | Modest |
| | Team Oriented | Categorization | Loyal |
| | Productive | Comprehension | Flexible |
| | Persistent | Planning | Independent |
| | | Decision making | |
| | | Problem solving | |
| | | Analysis | |
| | | Synthesis | |
| **Sociability** | **Rigorousness** | **Creativity** | **Custom** |
| Collaboration capability | Contingent error rate | Abstraction capability | Exterior hobby |
| Communication capability | Repeatable error rate | Imagination | Interior hobby |
| Extroversion | Error-correction capability | Analogy capability | Quietness |
| Introversion | Pinpoint capability | Curiousness | Activeness |
| Culture factor | Concentration capability | Design ability | Literature |
| Leadership | Logical inference capability | Hands-on capability | Vision |
| Group orientation | Reliability | Broad mind | |
| Organization capability | Precision | | |
| Concern of others | Perception | | |
| Dependability | Consistency | | |
| Compatibility | System | | |
| | Talent | | |

In their paper, Hillson and Webster (Hillson & Webster 2006) speak about the connection between emotions and risk behavior and try to show the relation between emotional literacy and work attitude.

Dhillon (Dhillon 2007) summarizes the important factors affecting the productivity of the individual work and names them "stressors". He categorized stressors into four types:

- Occupational change-related stressors
- Occupational frustration-related stressors
- Workload related stressors
- Miscellaneous stressors

He defines also different reasons for the occurrence of human errors: (Dhillon 2007)

> *''Poor training or skill, poor equipment design, complex task, poor work layout, high temperature or noise level in the work area, distraction in the work area, poor lighting in the work area, poorly written equipment operating and maintenance procedure,; improper work tools, poor verbal communication, poor motivation, crowded work space and poor management''.*

Although the book concerns transportation systems, to our opinion all these factors can be applied also to the software development process.

Dayer (Dayer 2007) summarizes the factors that influence the human reliability into two groups: internal and external. The internal is formed by the company working atmosphere and the external by the individual personal life. Internal factors are, for example, trust and working climate while external factors refer to family, health and the Maslow's pyramid of needs (Maslow 1987).

Islam and Dong (Islam & Dong 2008) summarize the human risk factors as follows: ''personal competency, experience and leadership, team performance, availability of skilled personnel, commitment, personnel loyalty and different specific working skills.''

Yanyan and Renzuo (Yanyan & Renzuo 2008) explain the psychological background of human behavior as a mixture of human knowledge, emotion and intention. They try to find the relationship between software engineering and knowledge and at the same time to include the human factors that influence this knowledge.

Analogically to Dayer, Flouris and Yilmaz (Flouris & Yilmaz 2010) build a framework for human resource management where they divide the human characteristics into internal and external influenced ones. We can see below the list with the internal and external performance influencing factors.

*Internal Performance Influencing Factors* (Flouris & Yilmaz 2010)

•Emotional state
• Intelligence
• Motivation/attitude
• Perceptual abilities
• Physical condition
• Sex differences
• Skill level
• Social factors
• Strength / endurance
• Stress level
• Task knowledge
• Training/experience

*External Performance Influencing Factors* (Flouris & Yilmaz 2010)

• Inadequate workspace and layout
• Poor environmental conditions
• Inadequate design
• Inadequate training and job aids
• Poor supervision

Another taxonomy that we will consider is that from Kim and Jung (Kim & Jung 2003). They performed a study over 18 performance shaping factor taxonomies and summarized the human factors into the Table 3 that we can see below.

**Table 3 Kim and Jung's Human Factor Taxonomy (Kim & Jung 2003)**

| **Subgroup** | **Detailed items** | |
|---|---|---|
| **Cognitive characteristics** | **Cognitive states**<br>- attention<br>- intelligence<br>- skill level<br>- knowledge<br>- experience<br>- training | **Temporal cognitive states**<br>- memory of recent actions<br>- operator diagnosis<br>- perceived importance<br>- perceived consequences<br>- operator expectations<br>- confidence in diagnosis<br>- memory of previous actions |
| **Physical and psychological characteristics** | **Physical states**<br>- gender/age<br>- motor skills<br>- physical disabilities<br>- impediment<br>- clarity in speaking<br>- fatigue/pain<br>- discomfort<br>- hunger, thirst | **Psychological States**<br>- emotion/feeling<br>- confusion<br>- task burden<br>- fear of failure/consequences<br>- high jeopardy risk |
| **Personal and Social Characteristics** | **Personal**<br>- attitude<br>- motivation<br>- risk taking<br>- self-esteem<br>- self-confidence<br>- sense of responsibility<br>- sensation seeking<br>- leadership ability<br>- sociability<br>- personality<br>- anticipation | **Social**<br>- status<br>- role/responsibility<br>- norms<br>-attitudes, influenced by other people |

## 2.5 Summary over the Human Factors

The conducted overview of the scientific work over human factors in the software process as $HF_{IT}$ has few different perspectives:

- We have slips and mistakes occurring in everyday human work including their base (e.g. skill, rule or knowledge-based).
- Then we have malfunctions and their relation to the behavioral model of the human being with regards to performing or not a certain task.
- The connection between emotions and risk behavior is clearly recognized and different stressors that influence the people are categorized.
- We have different levels of failures and different factors that influence the human actions.
- We have observed different types of frameworks and taxonomies that list all different personal characteristics that influence the working process.

Having all this in mind we can say that there exist a lot of scientific attempts to connect the human behavior with the conducted mistakes in the work process, but nobody has tried so far to observe the personal traits and their influence over the individual's work performance. By personal traits, we understand the individual's characteristics that are important for every employee and that influence the working process as well as the occurrence of mistakes or different problems. Based on this, we will try to find the most important human features that affect the work quality in the software development process, to evaluate the most critical of them and to build a prediction model of the human performance.

We can say that all these different types of human factors are actually the human risks in the software development process which we have to cope up with (Neumann et al, 2010 a) (Georgieva et al, 2010 b). In order to be able to manage with the different types of slips, mistakes and errors we have to manage first the factors that cause them. We will visualize this in the following way.

From our research we can say that the human risk factors *HRF* can be divided in the following groups:

- Cognitive human risk factors $HRF_{cog}$,
- Physical human risk factors $HRF_{phys}$,
- Personal human risk factors $HRF_{pers}$,
- Social human risk factors $HRF_{social}$.

When we try to evaluate them in the software development process we have to take them as a whole but we can say that the different factors are connected with the variety of roles and their responsibilities or the involvement in the IT process.

Because of this we can establish the following relations:

$$personnel_{IT} = \{P_{dev}, P_{sys}\} \quad (2.4)$$

$$HF_{IT}: personnel_{IT} \times processInvolvement \times role_{IT} \rightarrow HF_{IT}$$

$$processInvolvement = \{ f(P_{dev}) \cup f(P_{sys}) \}$$

$$HF_{IT} = \{attention, communication, competence, concentration, cooperation, hardworking, intelligence, self\text{-}management, talkativeness, understanding, creativity, tolerance, positive, knowledge, motivation\}$$

where $f$ denotes any team and/or business aspects in concrete industrial environments. Note, that the different roles as so-called $role_{IT}$ we will consider later. Addressing risk implications, we can characterize:

$$HRF_{IT}: HF_{IT} \times processInvolvement \times humanRisks_{IT} \rightarrow personnelRisks \quad (2.5)$$

$$humanRisks_{IT} = \{ errors_{IT}, violations_{IT}, failures_{IT} \}$$

$$errors_{IT} = \{skillBasedErrors, decisionErrors, perceptualErrors, knowledgeBasedErrors\}$$

$$violations_{IT} = \{trainingRules, qualifications, socialFactors\}$$

$$failures_{IT} = \{unsafenessTasks, performanceSlips, organziationalMistakes\}$$

and furthermore

$$HRF_{IT} = \{ HRF_{IT}^{cog}, HRF_{IT}^{phys}, HRF_{IT}^{pers}, HRF_{IT}^{social} \}$$

$$HRF_{IT}^{cog} = \{ attention, intelligence, skillLevel, knowledge, experience \} \quad (2.6)$$

$$HRF_{IT}^{phys} = \{ gender, age, motorSkills, physicalDisabilities, fatigue, discomfort, impediment \}$$

$$HRF_{IT}^{pers} = \{ attitude, motivation, selfEsteem, selfConfidence, riskTaking, sensationSeeking, leadershipAbility, sociability, anticipation \}$$

$$HRF_{IT}^{social} = \{ status, role, responsibility, norms, attitudes \}$$

In following we will consider the $HF_{IT}$ in general including their exploration for risk situations (as $HRF_{IT}$ ) or in a positive manner as reasonable characteristics for IT processes.

# 3 Chapter - Software engineering, team and responsibilities

Our research is enclosed in the world of the software engineering and because of this we will give a short explanation of its' main parts in the following chapter. We will have a look over the software process, product and resources and will try to distinguish the importance of the human performance inside. Later on we focus on the software team roles and their responsibilities and describe them in order to understand the importance and the complexity of the human being in the software engineering process. This part ends with summarizing the personal characteristics for the different roles, which is the input for the further research in the next chapter.

## 3.1 The Software Engineering Background

### 3.1.1 Software Engineering characterization

Basically, the software engineering can be defined with the following classical IEEE description (IEEE 1990) that is:

> *„Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. "*

This definition leads us to the simple visualization of the software engineering components in the following manner (Dumke 2003) (Marciniak 1994) (Pfleeger 1998) (Dumke et al, 2010 ) (Georgieva et al, 2010 c).

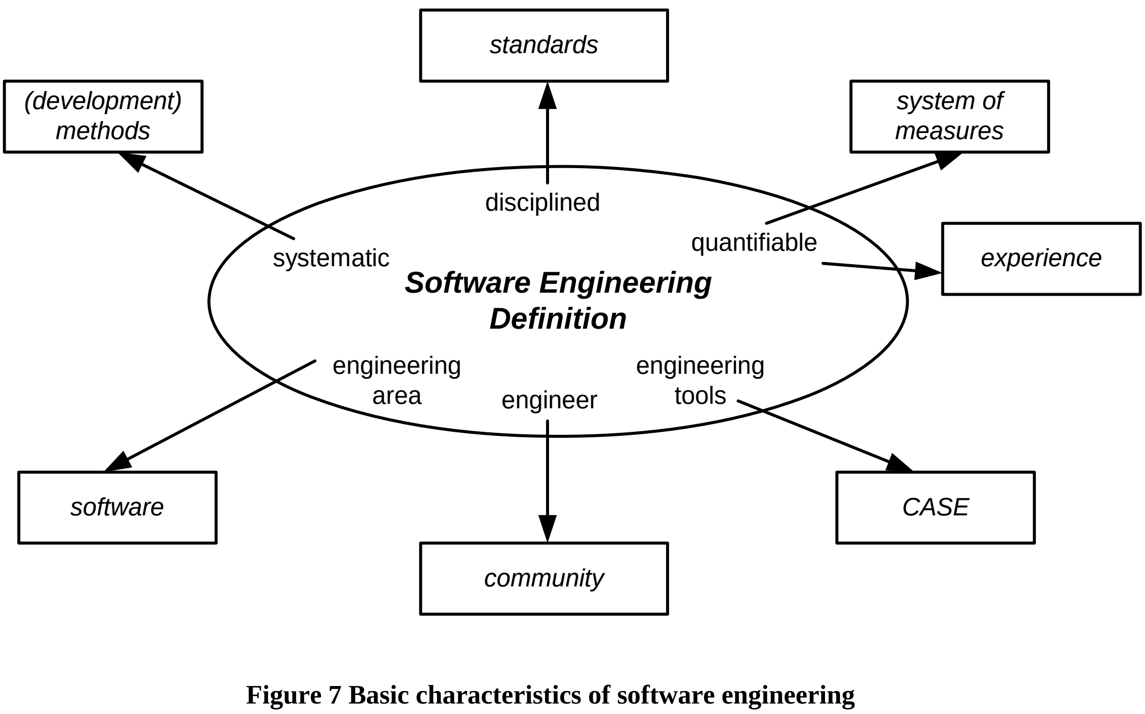



**Figure 7 Basic characteristics of software engineering**

Considering this characterization, we can formulate in the following simple structure of the software engineering *SE* area as a system in general: (Skyttner 2005)

$SE = (M_{SE}, R_{SE}) = (\{SE\text{-}Methods, CASE, SE\text{-}SystemOfMeasures^{1}, SE\text{-}Standards,$
$SE\text{-}SoftwareSystems, SE\text{-}Experience, SE\text{-}Communities\}, R_{SE})$

(3.1)

where $R_{SE}$ represents the set of all relations between the elements of the set $M_{SE}$ where the elements of $M_{SE}$ mean in detail:

*SE-Methods:* "Structured approaches to software development, which include system models, notations, rules, design advice and process guidance." (Sommerville 2008)

*CASE:* (Computer-Aided Software Engineering) "Software systems which are intended to provide automated support for software process activities." (Sommerville 2008)

*SE-SystemOfMeasures:* A set of metrics and measures in order to measure and evaluate all aspects, components and methodologies of the software engineering areas. (Zuse 1998) (Dumke et al, 2009 a) (Georgieva et al, 2009 e) (Dumke et al, 2009 b) (Dumke et al, 2008)

*SE-Standards:* The software engineering standards are a set of rules and principles as a foundation of control and examination of components achieving special defined characteristics certified by a consortium like IEEE or ISO. (Dumke 2003) (Georgieva et al, 2008)

*SE-SoftwareSystems:* A software system respectively a software product "is a purposeful collection of interrelated components that work together to achieve some objectives" and requirements. It includes the computer programs and the associated documentation. (Sommerville 2008)

*SE-Experience:* The experience summarizes the general aspects of laws, principles, criterions, methodologies and theories in software engineering in the different forms of aggregation, correlation, interpretation and conclusion based on a context-depended interpretation. (derived from (Davis 1995))

*SE-Communities:* The software engineering community involves people, organisations, events and initiatives in which interpersonal relationships are an integral part, considering aspects or paradigms in software engineering. (Figallo 1998)

Based on (3.1) we can formulate the following examples, components and elements of $R_{SE}$:

- The process of producing new or extended experience in software engineering:

  $r_{SE}^{(SE-Experience)} \in R_{SE}$: $SE\text{-}Methods \times CASE \times SE\text{-}SoftwareSystems \rightarrow SE\text{-}Experience$

(3.2)

[1] We use this kind of notification adapted from the OO area for more mnemonics.

- The general activities in order to define new standards in the SE:

$r_{SE}^{(SE-Standards)} \in R_{SE}$: *SE-Methods* × *SE-SoftwareSystems* × *SE-Communities* → *SE-Standards* (3.3)

- The process of extension the set of measures during the software development, maintenance or application :

$r_{SE}^{(SystemOfMeasures)} \in R_{SE}$: *SE-Methods* × *SE-SoftwareSystems* × *systemOfMeasures* → *systemOfMeasures*

(3.4)

- The process of risk management:

$r_{SE}^{(RiskManagement)} \in R_{SE}$: *SE-RiskAssessment* × *SE-RiskControl* → *RiskManagement* (3.5)

- The characterization of the software quality personnel:

$r_{SE}^{(RiskCommunity)} \in R_{SE}$: *SE-Communities* × *systemOfRiskMeasures* × *RiskManagement* → *RiskMeasurementStaff* (3.6)

### 3.1.2 The Software Product

The main intention of software engineering is to create/produce software products with a high quality for the customers. A software systems or software product *SP* was developed by the software process/development *SD* and is based of the supporting resources *SR*.

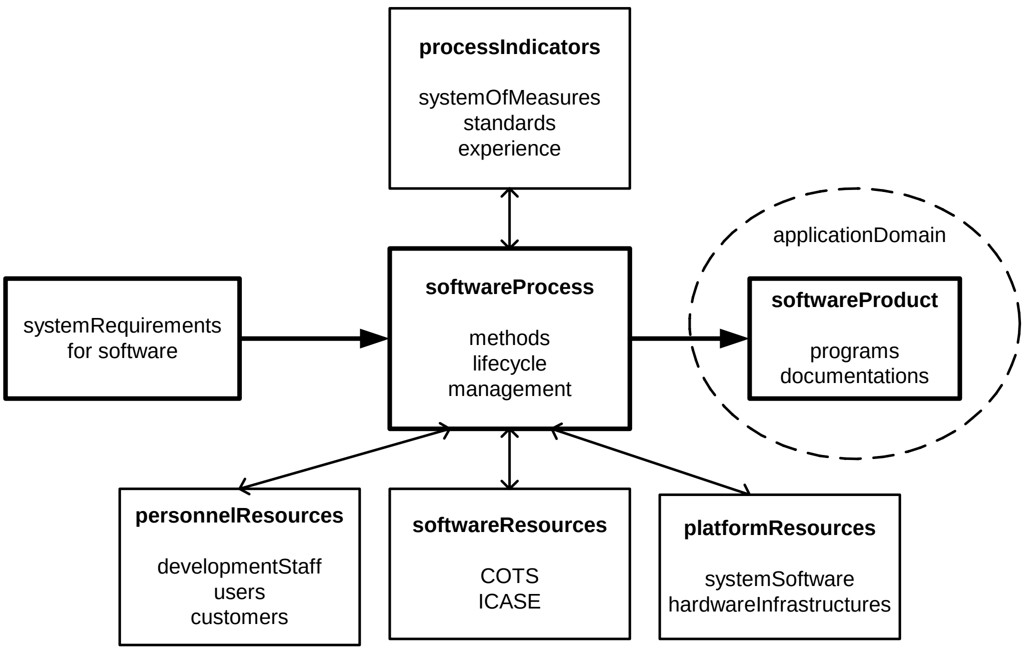



**Figure 8 The general software development process**

At first, we will define the software product as a (software) system as:

$$SP = (M_{SP}, R_{SP}) = (\{programs, documentations, data\}, R_{SP}) \quad (3.7)$$

where the three sets are divided in the following elements or components (without achieving the completeness)

$$programs \subseteq \{sourceCode, objectCode, template, macro, library, script, plugIn, setup, demo\} \quad (3.8)$$

$$documentations = \{userManual, referenceManual, developmentDocumentation\}$$

$$data = \{singleData, eventData, sensorData, dataBases, dataWarehouses, dataInfrastructures, knowledge\}$$

$$dataRisks = \{missing, incorrect, incomplete, not\ synchronized, misleading\}$$

and $R_{SP}$ describes the set of the relations over the *SP* elements.

The given subsets could be described in the following:

$$developmentDocumentation = \{documentationElements\} = \{productRequirements, productSpecification, productDesign, implementationDescription\} \quad (3.9)$$

$$documentationElements \subseteq \{model, chart, architecture, diagram, estimation, review, audit, verificationScript, testCase, testScript, pseudoCode, extensionDescription, qualityReport\}$$

$$productRequirements = systemRequirement \subseteq \{functionalRequirements, qualityRequirements, platformRequirements, processRequirements\}$$

$$functionalRequirements \subseteq \{execution, mapping, information, construction, controlling, communication, learning, resolution, cooperation, coordination\}$$ [2]

$$qualityRequirements \subseteq \{functionality, reliability, efficiency, usability, maintainability, portability\}$$ [3]

$$platformRequirements \subseteq \{systemSoftware, hardwareComponent, hardwareInfrastructure, peripheralDevice, host\}$$

$$processRequirements \subseteq \{developmentMethod, resources, cost, timeline, milestone, criticalPath, developmentManagement, lifecycleModel\}$$

A simplified view of the software product aspects during the development and application that must be defined through the product requirements can be seen on the following figure.

[2] The kind of the functional requirements depends on the kind of the software system which we characterize.
[3] This set of quality characteristics is related to the ISO 9126 product quality standard.

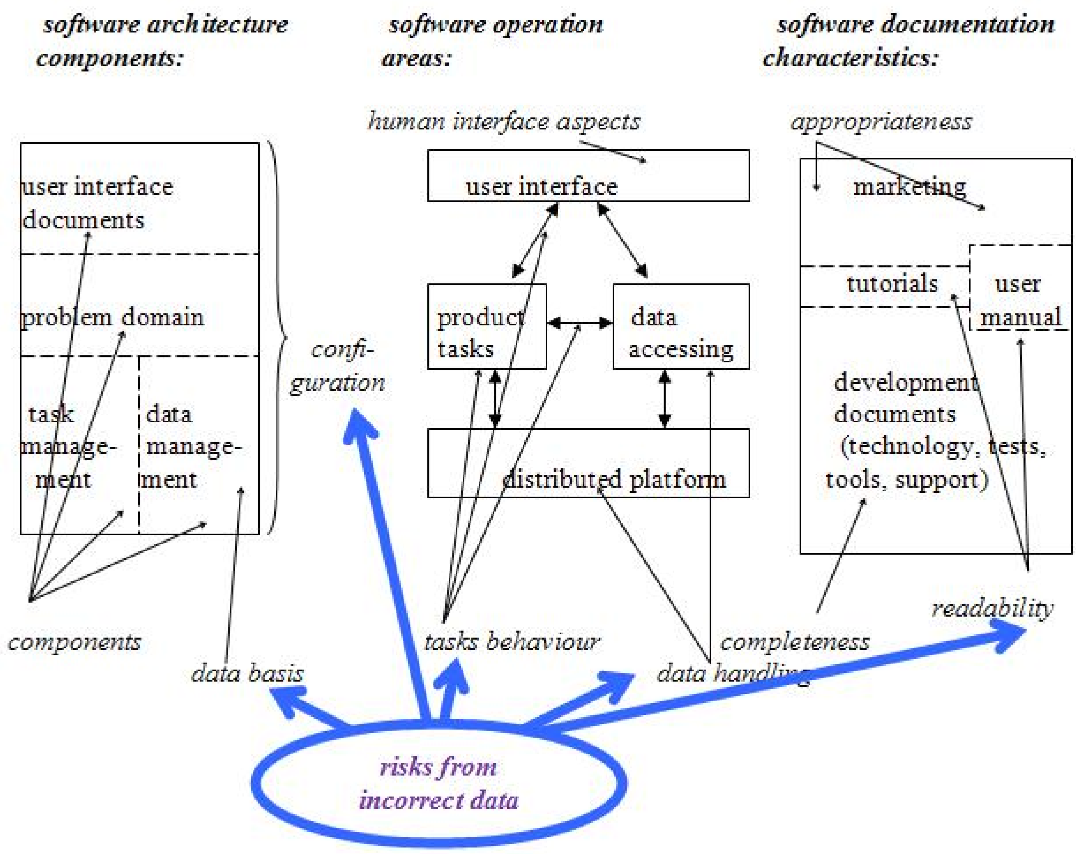



**Figure 9 Simplified visualization of the product characteristics and risks involvements**

This visualization could help us for further investigations of the detailed component and aspects of the software product. Here, we can define a software product as a software system as following (Chung et al. 2000) (Dumke 2003) (Horn & Reinke 2002) (Marciniak 1994) (Maciaszek 2001) (Mikkelsen & Phirego 1997).

$$SE\text{-}SoftwareSystems \subseteq \{informationSystem, constructionSystem, embeddedSystem, communicationSystem, distributedSystem, knowledgeBasedSystem\} \quad (3.10)$$

Some of the examples of the relations in $R_{SP}$ could be derived as following:

- The process of the software testing on some software product components, examples - (Farooq et al, 2008 a) (Farooq et al, 2008 b):

$$r_{SP}^{(test)} \in R_{SP}: sourceCode \times verificationScript \times testScript \rightarrow testDescription$$

(Farooq et al, 2010) (3.11)

- The elements of the product design considering the necessary components:

$$r_{SP}^{(design)} \in R_{SP}: architecture \times review \times template \times library \times pseudoCode \rightarrow productDesign \quad (3.12)$$

- A special kind of a programming technique could be defined as following:

$$r_{SP}^{(programmingTechnique)} \in R_{SP}: template \times macro \rightarrow sourceCode \quad (3.13)$$

- The process of the software testing on some software product components:

$$r_{SP}^{(implementation)} \in R_{SP}: coding \times unitTest \times integrationTest \rightarrow implementation \quad (3.14)$$

- The process of risk identification:

$$r_{SP}^{(riskIdentification)} \in R_{SP}: dataRisks \times applicationAnalysis \rightarrow riskIdentification \quad (3.15)$$

The following figure summarizes the components and elements of the software product described in the text above.

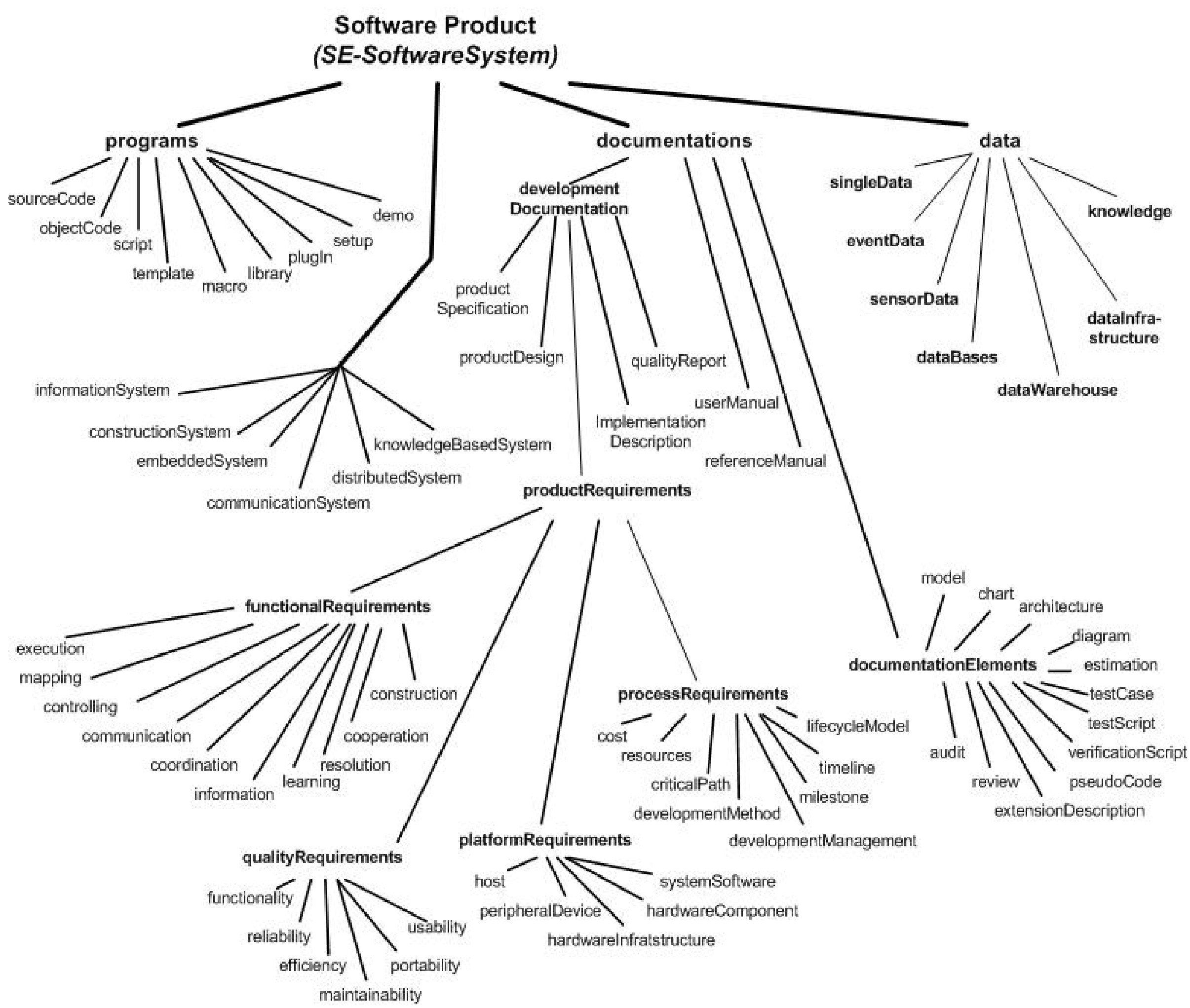



**Figure 10 Components of the software product**

### 3.1.3 The Software Development Process

Now, we will define the *software development process SD* itself (note, that the concrete software process is known as *software project*). Some special software enterprise applications can be seen in (Neumann et al, 2010 b) (Asfoura et al, 2011). At the beginning we will show the general process aspects in the following Figure 11.

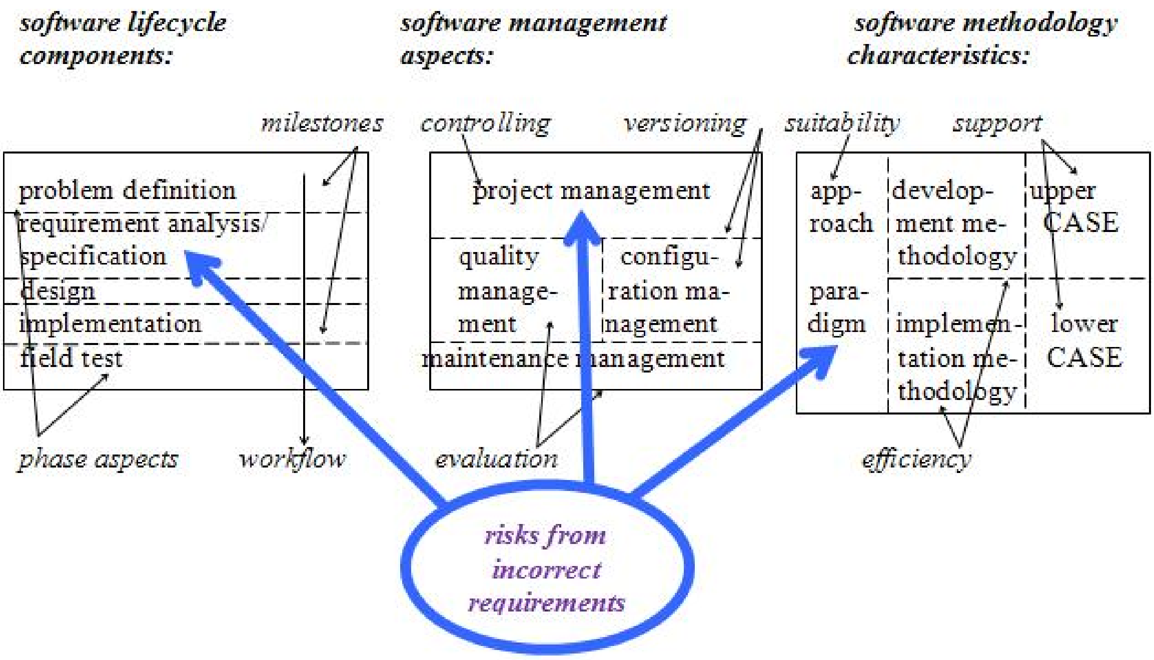



**Figure 11 Simplified visualization of the process characteristics and the risks involvements**

So, we can define the software process *SD* as following (including the essential details of every development component)

$$SD = (M_{SD}, R_{SD}) = (\{developmentMethods, lifecycle, softwareManagement\} \cup M_{SR}, R_{SD}) \quad (3.16)$$

$$developmentMethods \subseteq \{formalMethods, informalMethods\} = SE\text{-}Methods$$

$$formalMethods \in \{CSP, LOTOS, SDL, VDM, Z\}$$

We can see a plenty of "classical" informal development methods (Günther et al, 2011) as structured/procedural methods *SAM*. Actually, the informal methods are based on the objects *OOSE,* the components *CBSE,* the agents *AOSE* or the services SOSE (Neumann et al, 2011 a) (Neumann et al, 2011 b). Therefore, we can define:

$$informalMethods \in \{SAM, OOSE, CBSE, AOSE, SOSE\} \quad (3.17)$$

and especially

$SAM \in \{SA/SD, Jackson, Warnier, HIPO\}$

$OOSE \in \{UML, OMT, OOD, RDD, Fusion, HOOD, OOSA\}$

$CBSE \in \{DCOM, EJB, CURE, B\text{-}COTS, SanFrancisco\}$

$AOSE \in \{AAII, AUML, DESIRE, IMPACT, MAS, MaSE, MASSIVE, SODA\}$

$SOSE \in \{SOA, GRID, WebServices, Cloud\}$

The life-cycle aspects could be explained by the following descriptions:

$$lifecycle = \{lifecyclePhase, lifecycleModel\} \quad (3.18)$$

$$lifecyclePhase \in \{problemDefinition^{4}, requirementAnalysis, specification, design, implementation, acceptanceTest, delivering\}$$

$$lifecycleModel \in \{waterfallModel, Vmodel, evolutionaryDevelopment, prototyping, incrementalDevelopment, spiralModel, \ldots, winWinModel\}$$

$$requirementsRisks = \{incomplete, unrealistic, subjective, dependability, dynamic, incompatible, not\ measurable\}$$

Finally, the software management component of the $M_{SD}$ could be described in the following manner:

$$softwareManagement = developmentManagement \subseteq \{projectManagement, qualityManagement, configurationManagement, riskManagement\} \quad (3.19)$$

Note that the software development process (Dumke et al, 2009 c) could be depended or addressed to a special kind of a software system. Hence, we can make the following characterization:

$$SD_{informationSystem} \neq SD_{embeddedSystem} \neq SD_{distributedSystem} \neq SD_{knowledgeBased\ System} \quad (3.20)$$

- The process of risk management on a particular product: (Boehm 1991)

$$r_{SP}^{(riskManagement)} \in R_{SP}: riskIdentification \times riskAnalysis \times riskPrioritization \times riskMgmtPlanning \times riskResolution \times riskMonitoring \rightarrow riskManagement \quad (3.21)$$

Further, some of the examples of the relations in $R_{SD}$ could be derived in the following way:

- The process of building an appropriate life-cycle model:

$$r_{SD}^{(lifecycle)} \in R_{SD}: lifecyclePhase_{i_1} \times \ldots \times lifecyclePhase_{i_n} \rightarrow lifecycleModel \quad (3.22)$$

[4] Problem definition is a verbal form of the defined system or product requirements.

- The defining of software development based on the waterfall-model:

$$r_{SD}^{(waterfallRisks)} \in R_{SD}: problemDefinition \times specification \times design \times implementation \times acceptanceTest \times riskManagement \rightarrow waterfallModel \quad (3.23)$$

- The defining of software development based on the V-modell:

$$r_{SD}^{(VmodelRisks)} \in R_{SD}: (problemDefinition, softwareApplication, riskManagement) \times (specification, acceptanceTest, riskManagement) \times (design, integrationTest, riskManagement) \times (coding, unitTest, riskManagement) \rightarrow Vmodel \quad (3.24)$$

The characterization of the tool-based software development based on UML:

$$r_{SD}^{(UMLdev)} \in R_{SD}: UML \times developmentEnvironment_{UML} \times systemOfMeasures_{UML} \times experience_{UML} \times standard_{UML} \rightarrow developmentInfrastructure_{UML} \quad (3.25)$$

Finally, the components and aspects of the software engineering process are shown in the following figure.

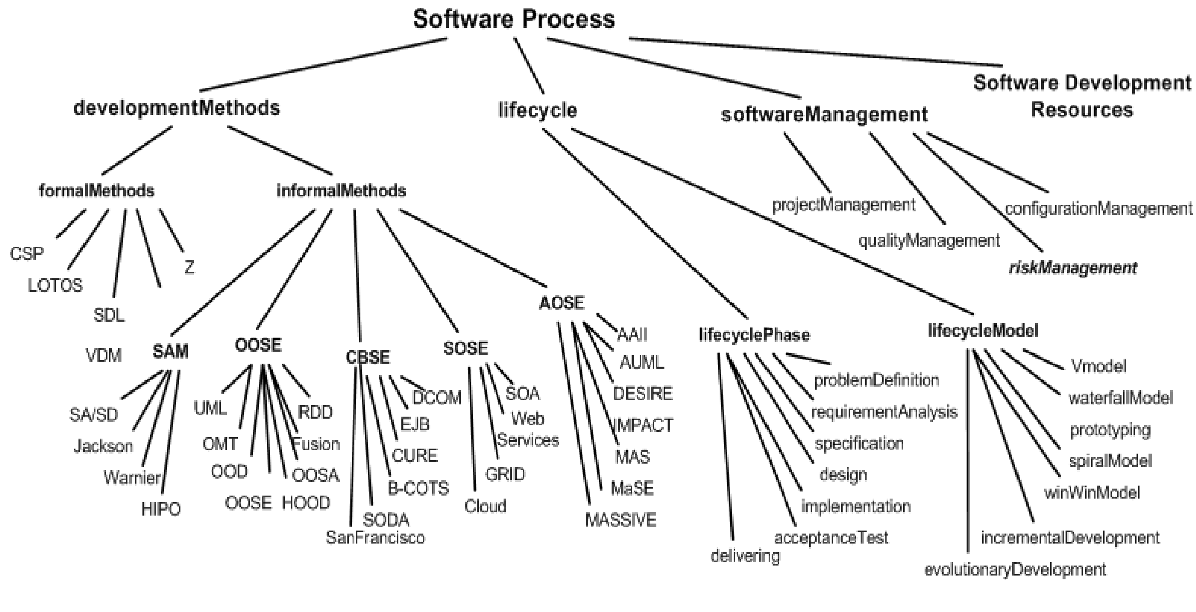



**Figure 12 Components of the software engineering process**

The components of the Risk Management that are an unavoidable part of the software process are shown here. We will go into some more details about the project risk management, because as we have already said: the concrete software process is known as *software project.*

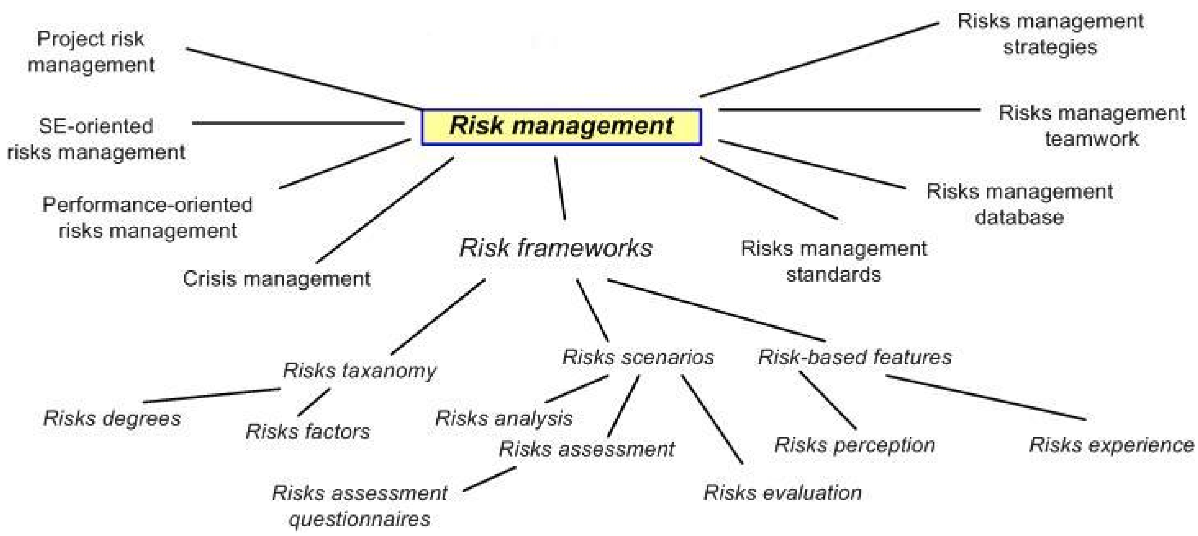



**Figure 13 Components of the risk management**

The software project risks as a part of the development process can be divided into different groups of risks, as follows: (Gaulke 2002)

*Business Focus Group Risks:*

Here are grouped the risks connected with the weak points in the matching between the project and the business goals and requirements of the company and also the external risks. The risk in the unsupported from the Business IT-project is the problem that some specific parts of the project may not have enough resources and then they won't have the possibility to be correctly developed.

*Stability of the Organization Risks:*

Changes in the company organization can be critical for the project. This could mean a change in the resources or even closure of the project. The restructuring of an organization because of extern circumstances or intern for example: new business field or efficient control, means extreme danger for the IT-project. The instability and the changes can have critical influence on the employees' motivation and this can be the point that brings a project to the end or not.

*Dynamic of the market-place Risks:*

The risk in the dynamic market-place is that in a case of change it could be that the project is not relevant any more or should be entirely changed. This leads to extreme lost of money and time and because of this it is very important to start with rich analysis of the market-place in order to be sure that the IT-project will be successful.

*Criticality of the IT-System Risks:*

The risk of implementing systems with high-criticality is that the expected security, performance or some feature could fail. The criticality of a system can be connected with the special function that should be fulfilled, for ex. bank-transfer or military communication or also with the business-risk.

*Special Risks:*

These are external and unexpected for the project factors that have negative influence over it. For example financial risks, even liquidity, crises on the market-place, or reputation loss (loss of personnel) can lead to extremely heavy problems for a project.

The software project/process risk can be expressed in the following way:

$$r_{SD}^{(processRisk)} \in R_{SD} = businessFocus \times organizationStability \times marketDynamic \times systemCriticality \times specialRisk \rightarrow processRisks \quad (3.26)$$

### 3.1.4 The Software Development Resources

In order to develop a software product we need resources such as developer (software team), CASE tools and variants of hardware. Therefore, we define the software development resources *SR* as following

$$SR = (M_{SR}, R_{SR}) = (\{personnelResources, softwareResources, platformResources\}, R_{SR}) \quad (3.27)$$

where the software resources play a dual role in the software development: as a part of the final system (as *COTS* or *software components*) and as the support for the development (as *CASE* or integrated CASE as *ICASE*). The following figure shows a possible distribution of the different characteristics addressed to the main parts of the software development resources.

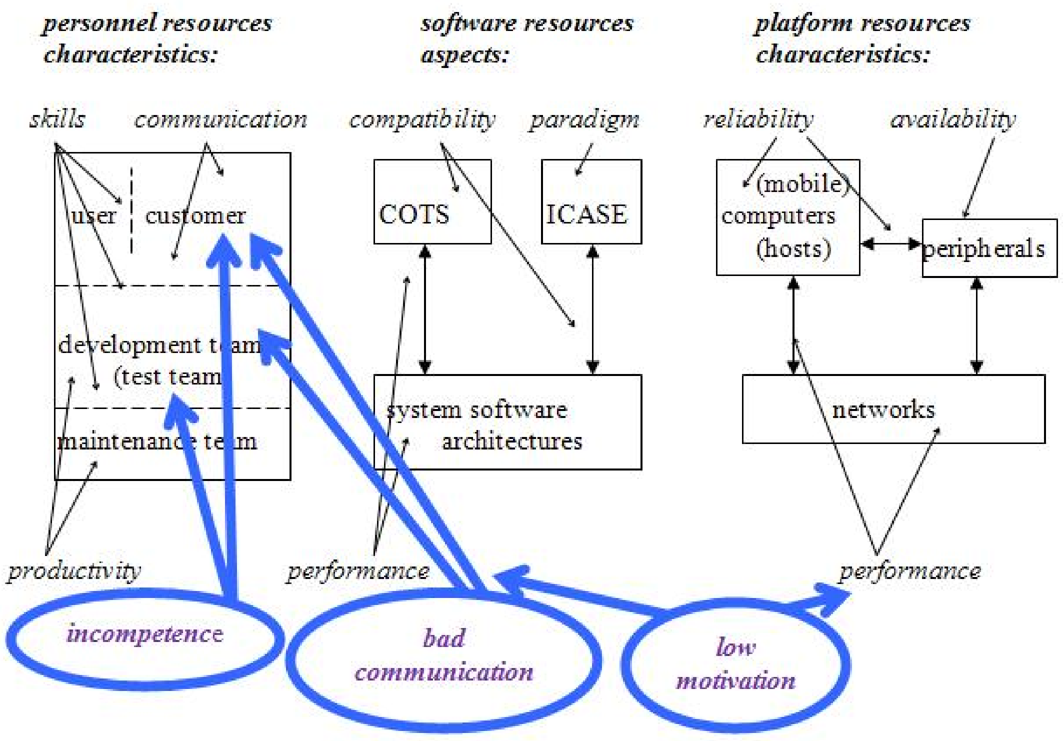



**Figure 14 Simplified visualization of the resources characteristics and the risks involvements**

We continue our definition as following:

$$softwareResources = \{COTS\} \cup \{ICASE\} \quad (3.28)$$

$$ICASE = CASE \cup CARE \cup CAME$$

where *CARE* stands for computer-aided reengineering and *CAME* means computer-assisted measurement and evaluation tools . Considering the WWW aspects and possibilities for software development infrastructures based on CASE environments, the set of CASE tools could be divided as following

$$CASE_{infrastructure} = \{ ( \{UpperCASE\} \cup \{LowerCASE\} )_{environment} \} \quad (3.29)$$

Further, we can define

$$UpperCASE = \{modellingTool, searchTool, documentationTool, diagramTool, simulationTool, benchmarkingTool, communicationTool\}$$

$$LowerCASE = \{assetLibrary, programmingEnvironment, programGenerator, compiler, debugger, analysisTool, configurationTool\}$$

Especially, we can describe the following types of software development resources as:

$$personnelResources = person_{IT} \cup person_{customer} \cup person_{applicatiuon} \quad (3.30)$$

$$person_{IT} = \{analyst, designer, developer, acquisitor, reviewer, programmer, tester, administrator, qualityEngineer, project\ leader, systemProgrammer, chiefProgrammer\}$$

$$person_{customer} = \{ stakeholder, manager, acquisitor \}$$

$$person_{application} = \{ user, operator, client, consumer \}$$

$$personnelRisks = HRF_{IT}$$

$$softwareResourcesRisks = \{notAvailability, highCosts, incomplete, incompatible, veryComplex, difficultyByChanges\}$$

$$hardwareResourcesRisks = \{lowPerformance, deadlocks, highCosts, incompatibility\}$$

and

$$SE\text{-}Communities = \{personnelResources, ITadministration, softwareUser, computerSociety\} \quad (3.31)$$

Accordingly, some of the examples of the relations in $R_{SR}$ could be derived in the following manner:

- The process of building an appropriate development environment:

$$r_{SR}^{(devEnv)} \in R_{SR}\text{: } ICASE \times platformResources \rightarrow developmentEnvironment \tag{3.32}$$

- The defining of software developer teams for the agile (for ex.) development:

$$r_{SR}^{(agile)} \in R_{SR}\text{: } programmer \times programmer \times customer$$
$$\rightarrow agileDevelopmentTeam \tag{3.33}$$

- The assessment of potential risks based on the personnel resources (see (2.4)):

$$r_{SR}^{(personnelRisks)} \in R_{SR}\text{: } HF_{IT} \times processInvolvement \times role_{IT} \rightarrow personnelRisks$$

- The assessment of the human performance:

  We have adopted the definition for Productivity (in our case synonym of Performance) ‘’Productivity is defined as output over input.’’ (Ebert & Dumke, 2007)

  Where ‘’Output can be:
  1. delivered source statements, function points, components, documents or artefacts.
  2. with a certain quality or complexity.
  3. in a certain environmental setting such as skills, pressure, tool support, computing platform, frequency of requirements changes,…
  4. having created application-domain and technical knowledge.’’ (Ebert & Dumke, 2007)

  And ‘’input is the way you create this output. It relates how well you are working.’’ Examples are:
  1. ‘’Productivity = adjusted size/effort. Adjusted size is the estimated effort based on history and constraints. Productivity is a normalization comparing estimated to actual effort.
  2. Productivity can be measured as a dimensionless indicator generated by an estimation method and tool, such as QSM SLIM, COCOMO or SPR Knowledge-Plan.
  3. Productivity can also be measured by comparing earned value with actual effort spent.‘’ (Ebert & Dumke, 2007)

  Having these explanations we have decided to use in our Performance Evaluation, three different components based on the Personal, Supervisor and Colleague Assessment based over the observed input-output dependence.

$$humanPerformance = \{HF_{IT}, softwareDevelopmentProcess\} \tag{3.34}$$

$$humanPerformanceEvaluation = \{personalAssessment, supervisorAssessment,$$
$$colleagueAssessment\}$$

$$r_{SR}^{(personalAssessment)} \in R_{SR}\text{: } person_{IT} \times assessment \times workingProcess$$
$$\rightarrow personalAssessment$$

$$r_{SR}^{(supervisorAssessment)} \in R_{SR}: person_{IT} \times supervisor \times assessment \times workingProcess \rightarrow supervisorAssessment$$

$$r_{SR}^{(colleagueAssessment)} \in R_{SR}: person_{IT} \times colleague \times assessment \times workingProcess \rightarrow colleagueAssessment$$

Now, we will summarize the different elements and components of the resources as the basics of the software development and maintenance in the following figure.

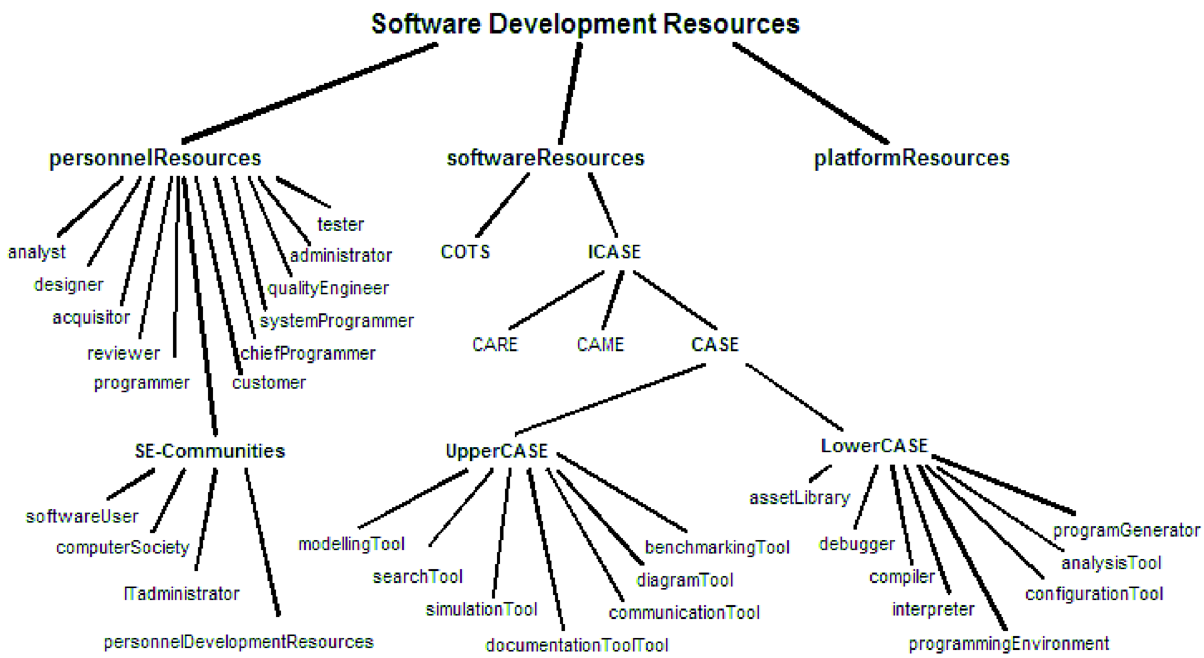



**Figure 15 Components of the software development resources**

### 3.1.5 The Use of the Software Product

After the software development, the software product goes in two directions: at first (the original sense of a software product) to the software application *SA*, second in the software maintenance *SM.* We will define here the different aspects:

$$SA = (M_{SA}, R_{SA}) = (\{applicationTasks, applicationResources, applicationDomain\} \cup M_{SP}, R_{SA}) \quad (3.35)$$

where

$$applicationTask \in \{delivering, operation, migration, conversion, replacement\}$$

$$applicationResources = \{applicationPlatform, applicationPersonnel, applicationDocuments\}$$

*applicationPersonnel* ⊆ {*customer, user, operator, administrator, consultant, trainer*}

*applicationDomain* ⊆ {*organisationalDocument, law, contract, directive, rightDocument*}

*applicationDocument* ⊆ {*userManual, trainingGuideline, acquisitionPlan, setup, damageDocument, troubleReport*}

The risks connected with the application Personnel in the process of use of the software product, can be summarized like the following:

$$riskInUse \subseteq \{lackOfExperience, lackOfResources, strongDependencies, lackOfUnderstanding, notFlexibleOrganization, lackOfGoalValidation, highSystemComplexity, badInformationStructure, lackOfData\} \quad (3.36)$$

Based on these definitions, some of the examples of the relations in $R_{SA}$ could be derived in the following manner:

- The process of the first introduction of the software product as delivery:

$$r_{SA}^{(delivery)} \in R_{SA}: SP \times trainer \times applicationPersonnel \times applicationPlatform \rightarrow delivery \quad (3.37)$$

- The defining of software migration based on essential requirements:

$$r_{SA}^{(migration)} \in R_{SA}: productExtension \times SP \times migrationPersonnel \rightarrow migration \quad (3.38)$$

- The characterization of software operation:

$$r_{SA}^{(operation)} \in R_{SA}: applicationPersonnel \times applicationPlatform \times SP \times user \rightarrow operation \quad (3.39)$$

- The defining of the outsourcing of the software operation by extern IT contractors:

$$r_{SA}^{(outsourcing)} \in R_{SA}: systemInputs \times contractors \times systemFeedback \rightarrow outsourcing \quad (3.40)$$

From all these relations can be summarized the source of risks for the software application (Georgieva et al, 2009 f) in the following manner:

$$r_{SA}^{(applicationRisk)} \in R_{SA}: deliveryRisk \times migrationRisk \times operationRisk \times outsourcingRisk \rightarrow applicationRisk \quad (3.41)$$

We can see all parts of the software product application in the following figure.

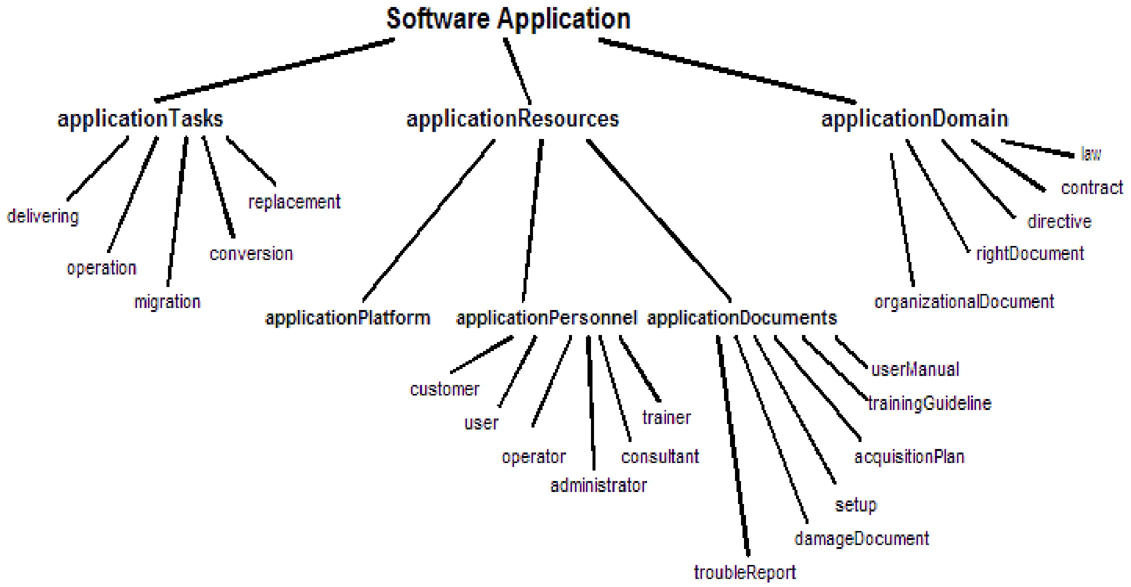



**Figure 16 Components of the software product application**

### 3.1.6 The Software Maintenance

The different aspects and characteristics of the software maintenance are summarized by the following formulas:

$$SM = (M_{SM}, R_{SM}) = (\{maintenanceTasks, maintenanceResources\} \cup SP) \qquad (3.42)$$

where

$$maintenanceTasks = \{extension, adaptation, correction, improvement, prevention\}$$

$$maintenanceResources = ICASE \cup \{maintenancePersonnel, maintenancePlatform\}$$

$$maintenancePersonnel = \{maintainer, analyst, developer, customer, user\}$$

Accordingly, some of the examples of the relations in $R_{SM}$ could be derived like follows:

- The process of building the extension activity of the maintenance:

$$r_{SM}^{(extension)} \in R_{SM}: SP \times functionalRequirements \rightarrow SP^{(extended)} \qquad (3.43)$$

- The defining of software correction:

$$r_{SM}^{(correction)} \in R_{SM}: SP \times qualityRequirements \rightarrow SP^{(corrected)}$$

- The defining of software adaptation:

$$r_{SM}^{(adaptation)} \in R_{SM}: SP \times platformRequirements \rightarrow SP^{(adapted)}$$

- The defining of software improvement:

$$r_{SM}^{(perform)} \in R_{SM}: SP \times performanceRequirements \rightarrow SP^{(improved)}$$

- The defining of software prevention:

$$r_{SM}^{(prevention)} \in R_{SM}: SP \times preventionRequirements \rightarrow SP^{(modified)}$$

- The characterization of a special kind of software maintenance as remote maintenance:

$$r_{SM}^{(remoteMaint)} \in R_{SM}: ICASE_{remote} \times maintenanceTasks \times maintenancePersonnel \rightarrow remoteMaintenanc \quad (3.44)$$

- The risk in the software maintenance can be summarized like:

$$r_{SM}^{(maintenanceRisk)} \in R_{SM}: extensionRisk \times correctionRisk \times adaptationRisk \times improvementRisk \times preventionRisk \times remoteMaintRisk \rightarrow maintenanceRisk \quad (3.45)$$

We can see the components of the software maintenance on the next figure.

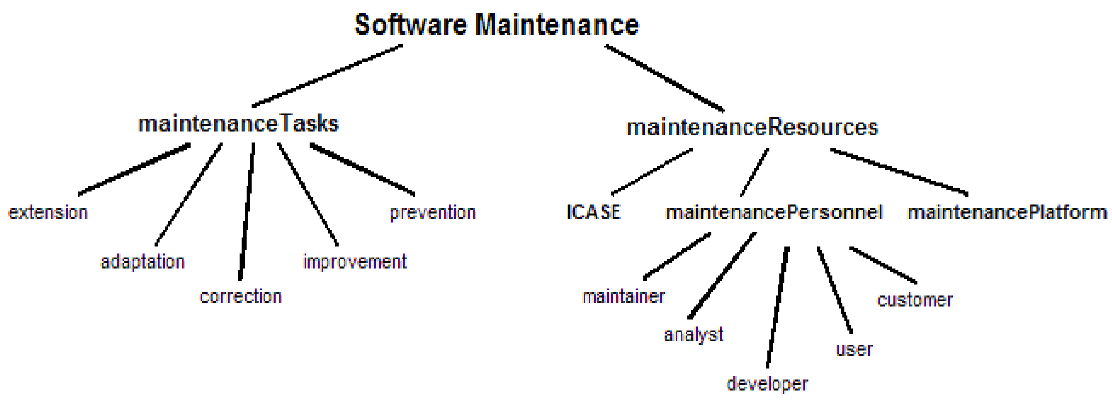



**Figure 17 Components of the software maintenance**

## 3.2 The Software Team

At the core of every software development process are the people. If the software development is considered as a project, then the people build the project successful or not. Independent of the methodology chosen for a particular project, a group of people called the project team is involved in it. The generalized roles involved in the software development process are provided in the following table.

**Table 4 General Roles in the Software Development (Kurble 2008), (Laporte et al. 2007), (Bogue 2005)**

| General Roles |
|---|
| Project Manager |
| Business Analyst |
| Software Architect |
| Team Leader |
| Software Developer |
| Quality Engineer |
| Software Tester |

In order to achieve the project goals, the project team has to be organized in a specific manner, called the project team structure. This structure is primarily a function of project resource ownership and project manager authority. Project manager's responsibility for achieving the project performance objectives must be supported by an appropriate level of authority to control project resource utilization, assign and manage project task performance, and enforce accountability of the project team members. Otherwise, the designated project leader is merely serving as a project coordinator or project report administrator and cannot reasonably be held responsible for project outcomes.

The software development process is executed with in the organizations. Each organization has its own organizational structure. So the project team structure depends on the organizational structure of the company in which the software is developed. Availability of resources, manager's authority, budget control and many more factors depend on the organization of the company. Therefore, the possible organizational structures are discussed in detail and the most appropriate organizational structure is specified for the software development.

### 3.2.1 Organizational Structures in the IT

An organizational structure is the "formal system of task and reporting relationships that controls, coordinates, and motivates employees so that they cooperate to achieve an organization's goals" (Kurble 2008). There are three basic types of organizational structures: (Heldman 2009) (PMI 2008)

- Functional Organization
- Projectized Organization
- Matrix Organization

3.2.1.1 Functional Organization

The functional organization, shown in the Figure 18, is an organization which is structured according to the functions such as analysis, design, implementation, testing and quality etc. Here software personnel are grouped by specialty, i.e. people with similar skills are placed in the same group. Each group has one head called the Functional Manager. Each employee has one clear superior. (Heldman 2009) Each group is managed independently and has a limited span of control. (Kerzner 2009)

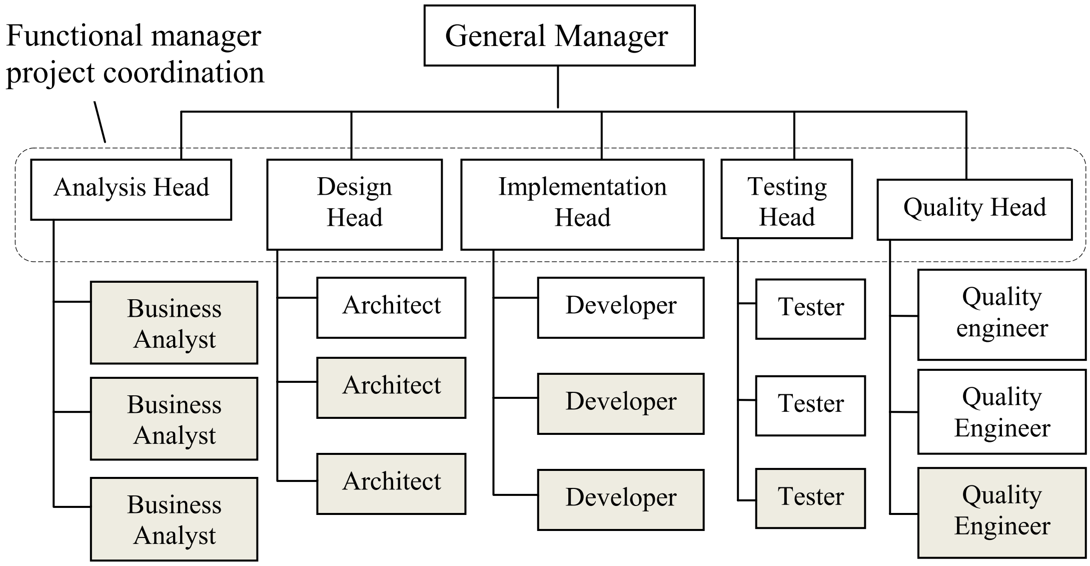



**Figure 18 Functional Organization. Gray boxes represent the people engaged in the same project (PMI 2008)**

Whenever a project has to be carried out in a functional organization, personnel from several functional areas work together. In this type of organization, a project manager is optional. Even if a project manager is assigned to a project, the project manager has little or no authority over project resources. Instead, the functional manager has complete authority over the project resources in a business unit.

The projects are typically undertaken in a divided approach (Heldman 2009) i.e. for a project in design phase includes the design department, will work on its portion of the project and then hand it off to the implementation department to complete its part and so on. Here a chain of command is followed. For example when questions about design arise in implementation phase then they are passed up the organizational hierarchy to the department head, who consults with the head of the design department.  The design department head then passes the answer back down the hierarchy to the implementation functional manager. In a real organization – in a multi-level hierarchy – the path upwards and downwards the organizational tree can be long and time-consuming.

Even though, the functional organizations have the advantage of being simple to understand with clear lines of command, it also has some disadvantages. The following are the advantages and disadvantages of a functional structure.

*Advantages*: (Kerzner 2009)

- ‘’Development and maintenance of technical competency in specialized fields
- Synergy among specialists
- Concentration on the objectives of the function
- Pursuing long-term development objectives
- Easy reconciliation of internal objectives
- Horizontal relations are clear
- Clear definitions of roles and responsibilities
- Efficiency improved by standardization
- Stability in interpersonal relations
- Well-defined career paths
- The possibility for organizational learning
- Easier control of quality and performance
- Flexibility and economy in the use of labor’’

*Disadvantages:* (Kerzner 2009)

- ‘’Filtered perception; lack of an overall view
- Difficulty in integrating several specialties; possible conflicts
- Difficulty in creating motivation for the project
- Risk of neglecting the aspects not related to the specialty
- Difficulty in making effective compromises between the variables quality-time-cost
- Nobody is exclusively responsible for project objectives
- Subordination of the managerial to the technical points
- Difficulty in adapting
- Difficulties in the internal circulation of information
- Lack of visibility for the client
- Limited development of management capabilities among the personnel’’

#### 3.2.1.2 Projectized organization

Projectized organizations (Heldman 2009) are almost the opposite of the functional ones. The idea behind them is to be loyal to the project manager and to organize the working process in the form of projects where all people are in project teams headed by a project manager to whom they report. Organizational resources are dedicated to projects and project-work. Figure 19 depicts a typical projectized organization.

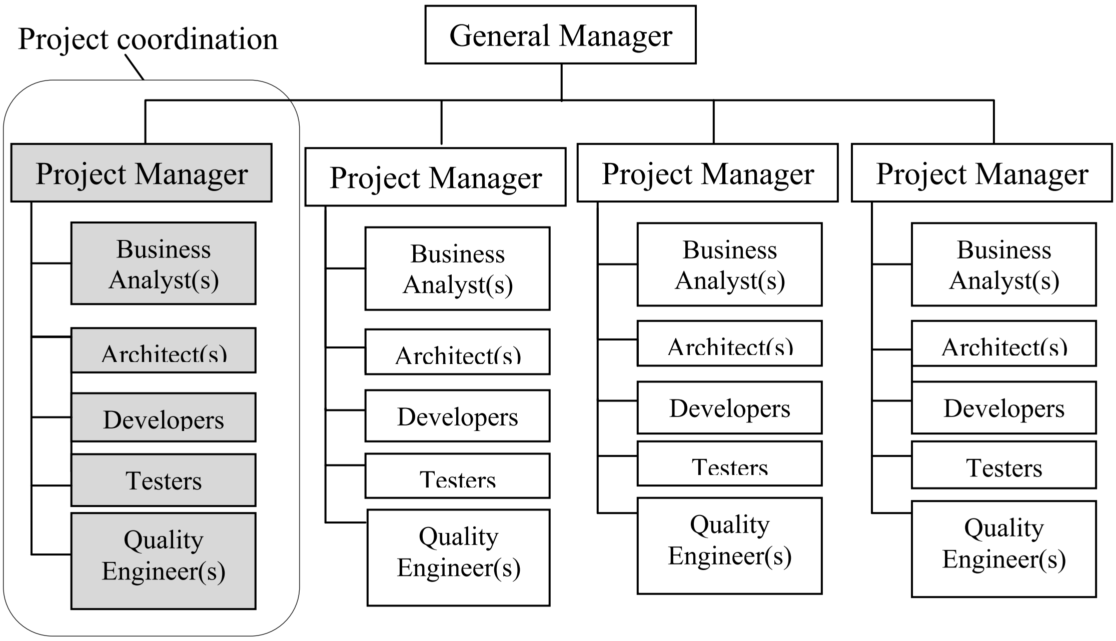



**Figure 19 Projectized Organization. Gray boxes represent staff engaged in project activities (PMI 2008)**

Project managers have the absolute power over the project in this structure and report directly to the General Manager. They are responsible for making decisions regarding the project acquiring and assigning resources and have the authority to choose and assign resources from other areas in the organization or from outside. (Heldman 2009) Project managers in all organizational structures are limited by triple constraints: project-scope, schedule and cost.

Project teams are formed from various specialists and are often co-located, which assures good communication. Motivation for project activities is high since the project is the main focus of the team.

Even though, it is a better organizational structure than the functional structure, it has some drawbacks. The following are the advantages and disadvantages of a projectized organization:

*Advantages*: (Kerzner 2009)

- ‘’Project managers have ultimate authority over the project
- Direct work for the project manager
- Strong communication
- Personnel demonstrate loyalty to the project
- Very rapid reaction time is provided
- Interface management becomes easier as unit size is decreased
- Team members are co-located’’

*Disadvantages*: (Kerzner 2009)

- ‘’Duplication of effort and resources
- A tendency to retain personnel on a project long after they are needed
- Technology suffers because without the functional groups, outlook of the future to improve company’s capabilities for new programs would be reduced

- Control of functional specialists requires top-level coordination
- Lack of opportunities for technical interchange between projects
- Lack of career continuity for project personnel’’

3.2.1.3 Matrix Organization

The matrix organizational form is an attempt to combine the advantages of the previous two structures. Here the project team members continue within their own functional groups, reporting to their usual managers for the purposes of career development and performance evaluation. (Heldman 2009) (Dinsmore 2010) By the matrix organizations the project managers can focus on the project work and the project team can focus on the project objectives without being distracted by the functional department. The project manager manages the project and the employees report to one functional manager and to at least one project manager.

Functional managers are concerned with the administrative duties and assign employees to the different projects and in the same time they maintain the projects’ quality. (Kerzner 2009) The functional managers have to assure a unified technical base that allows an exchange of information in every project and an awareness of the latest technical accomplishments in the industry. On the other hand, the project manager has total responsibility and accountability for the project success. The project managers are responsible for executing the project and assigning the tasks to the team members according to the project activities. The Figure 20 depicts the matrix organizational structure.

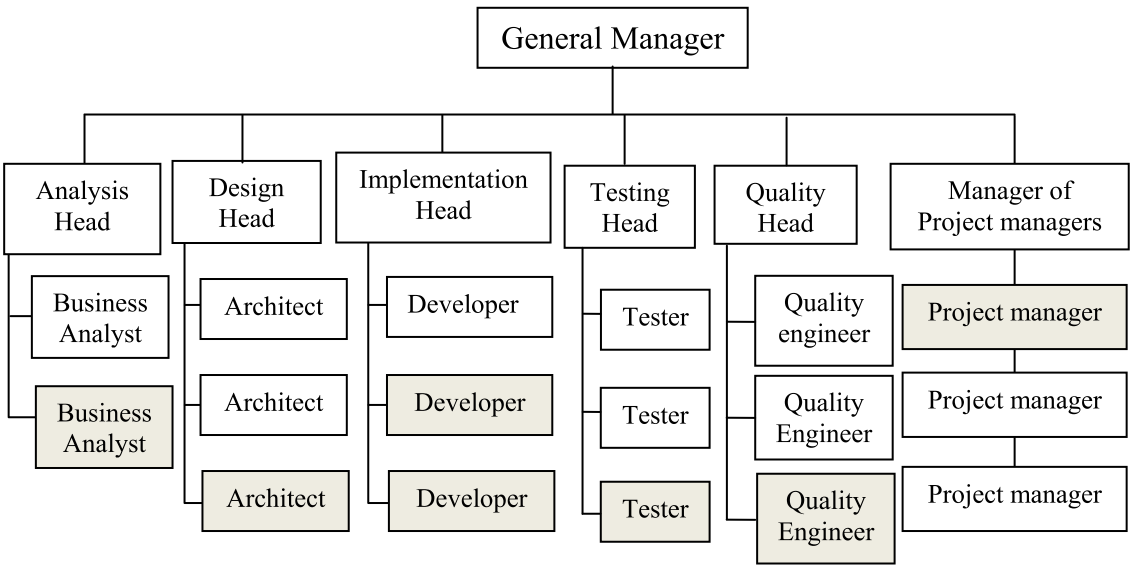



**Figure 20 Matrix Organizational Structure (PMI 2008)**

The gray color indicates the staff associated to a particular project manager.

Although matrix organizational structure is more beneficial than the other two structures, it has the following pros and cons.

*Advantages*: (Kerzner 2009)

- ‘’The project manager maintains maximum project control over all resources, including cost and personnel
- The project manager has the authority to commit the company resources mitigating conflicts with other projects
- Rapid responses are possible to changes, conflict resolution, and project needs
- The functional organization exist only as a support for the project
- Each person has a “home” after project completion.
- Because key people can be shared, the project cost is minimized. People can work on a variety of projects
- Conflicts are minimal, and those requiring hierarchical referrals are more easily resolved
- There is a better balance among time, cost and performance
- Authority and responsibility are shared
- A strong technical base can be developed, and much more time can be devoted to complex problem solving’’

*Disadvantages*: (Kerzner 2009)

- ‘’Multidimensional work flow
- Multidimensional information flow
- Dual reporting i.e. reporting to the functional and project manager
- Continuously changing priorities
- Management goals different from project goals
- Potential for continuous conflict and conflict resolution
- Each project organization operates independently. Care must be taken that duplication of efforts does not occur
- More effort and time are needed initially to define policies and procedures, compared to traditional form
- Functional managers may be biased according to their own set of priorities
- Balance of power between functional and project organizations must be watched
- Employees and managers are more susceptible to role ambiguity than in traditional form’’

In matrix organizations there exist different possibilities for the range of the organizational structure: we have weak, balanced and strong matrix.

In a strong matrix organization, the power is by the project managers, who take the most important decisions. Of course on the other end of the organizational structure spectrum is the weak matrix, where the functional managers have all the power and the project managers are just coordinators or expeditors.

In the middle is the so called balanced matrix organizational structure and it differentiates with the advantage of balancing between project managers and functional managers.’’ Each manager has responsibility for their parts of the project or organization, and employees get assigned to projects based on the needs of the project, not the strength or weakness of the manager's position.'' (Heldman 2009) Balanced matrix organization is shown in the Figure 21.

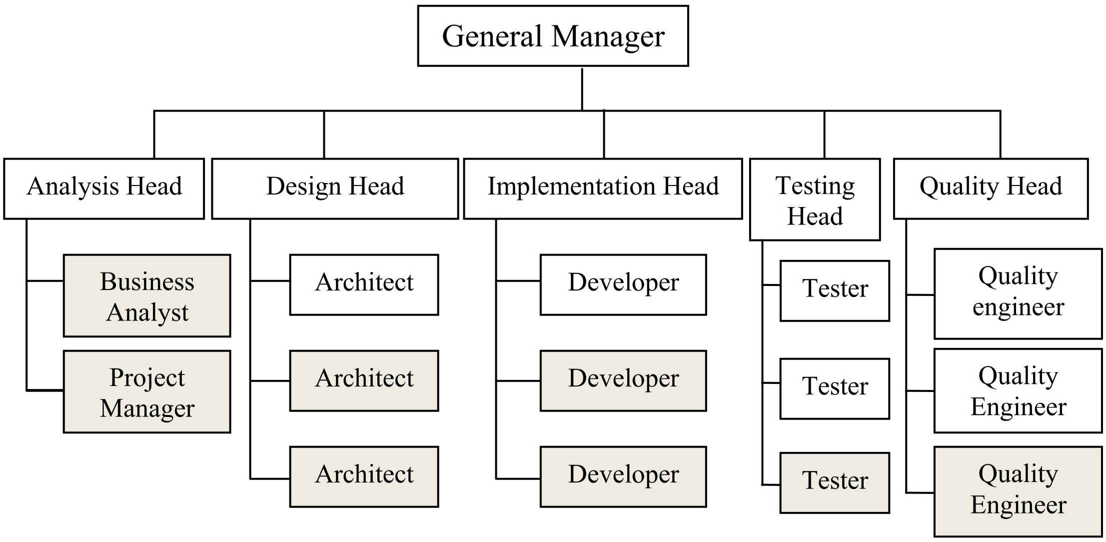



**Figure 21 Balanced Matrix Organization. Gray boxes represent staff engaged in project activities (PMI 2008)**

3.2.1.4 Organizational structure of a software company

Having discussed the possibilities for organizational structures and their pros and cons we will now observe the most suitable structure for a software company proposed in (Kurble 2008) - the strong matrix organization.

The functional areas of a software company are analysis, design, coding, testing etc. These functional areas can be arranged in a hierarchical manner and strong matrix organization is used in executing the software projects. The Figure 22 shows the typical organizational structure of a software company. We have the following departments: (Kurble 2008)

– Project management – project manager, assistant project manager and administrative personnel
– Sales and marketing – not visualized in our case, but usually a part of a software firma
– Analysis – requirements engineers or systems analysts performing requirements engineering
– Design – software architects developing the architecture of the system; class, database and GUI designers
– Implementation – java; database and GUI programmers
– Testing – staff performing module, integration and system testing
– Standards – a quality officer or assistant to ensure that software engineering standards are met.

These roles are the basic ones in the outcome of a software project. They are responsible for the success or failure of a project. Each role appears in some step of the software development life cycle and is assigned with particular responsibilities. In the next section, we will discuss the responsibilities of each role.

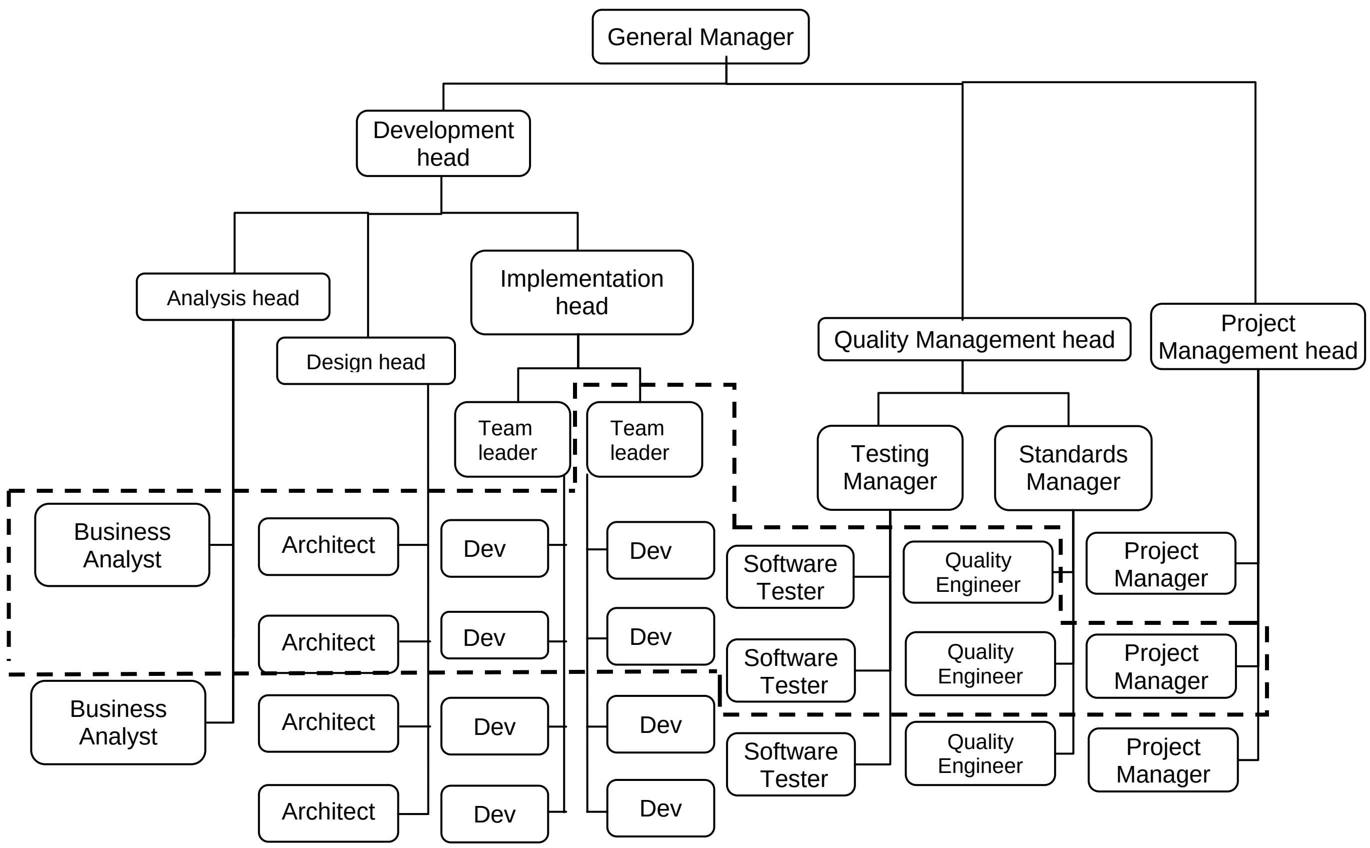



**Figure 22 Software Development Organizational Structure. The people in the dotted line indicate staff engaged for a project, Dev is the developer (PMI 2008)**

## 3.2.2 Software Roles and Responsibilities

### 3.2.2.1 Project Manager

The purpose of the project manager's role is to undertake the phases, activities and tasks within the specified time, cost and quality constraints to deliver the required software project outcome and achieve total customer satisfaction. ''A project manager's task is threefold: to supervise the team members, understand state of the art techniques, and make the software project successful.'' (Sodhi & Sodhi 2001)

The project manager is responsible for the controlling of the software development work from the initial beginning through to the end. This includes all software phases: planning, product design and development, implementation, administration, and setting and meeting of deadlines. (Desmond 2004) The project manager must have the following personal competencies and meet the following technical responsibilities for the successful outcome of the software project.

*Personal Competencies*: (Desmond 2004) (Sodhi & Sodhi 2001)

- ''Has good communication and managerial skills
- Able to lead, motivate and delegate proper responsibilities to team members
- Respects team members and has their respect
- Shares success with the team members
- Creates structured discipline
- Recognizes individual differences and takes advantage of them
- Understands the team members and creates effective communication
- Resolves conflicts and interpersonal issues promptly
- Open for new ideas
- Achieves the project goals within the established schedule and budget
- Establishes and meets real priorities and deadlines
- Is constantly in learning mode
- Has a structured roadmap for implementing change initiatives
- Has a habit of seeking improvements in all aspects of one's work''

*Technical Responsibilities*: (Desmond 2004) (Sodhi & Sodhi 2001)

- ''Detailed understanding of project planning and control techniques
- Ability to produce a detailed project plan, including a work breakdown structures, dependencies, resources and costs
- Optimizes the use of people and resources
- Knowledge of effective change management processes and procedures
- Constantly challenges established practices in order to improve them
- Has a complete understanding of the business and creates effective business plans
- Has knowledge of software development life cycle
- Knowledge of quality assurance and control techniques to ensure that quality targets and standards are met
- Identifies problems at an early stage and takes corrective actions
- Review or establish a hierarchy of objectives and identify higher-level project objectives
- Review the project documentation and project description that defines the authority, responsibility and relationships of the project manager, project staff, and functional departments
- Evaluate the probability of successful implementation and determine if some changes are needed
- Develop a plan to cope with potential problems caused by actors and factors especially those characterized by medium or high dependency, risk and control
- Lead the project team in reviewing the project documentation in order to reach a common understanding of objectives, deliverables, organizational structure etc,
- Review the proposed project implementation plan

- Verify resource commitment with the heads of the project related departments
- Prepare a presentation to management (project review) and negotiate it
- Plan inspection and acceptance procedures for the final deliverables
- Conduct final project review or audit
- Close out contracts and settle any outstanding disputes
- Close out all work orders and project accounts''

3.2.2.2 Team Leader

The team leader acts as a middle-point between the software architect and the developers. Depending on the project size, the team leader is responsible for extracting out details for a part of the architecture or the complete architecture and creating program specifications from which the developers work. Usually the team leaders were developers and have grown up in the hierarchy in the role of supervisors and guide the rest of the team during the software development process.

The team leader glues together the programs developed by the developers which forms a part or the whole architecture created by the software architect. In order to successfully lead a group of developers, a team leader should possess the following personal competencies and fulfill the following technical responsibilities.

*Personal Competencies*: (Humphrey 2005) (Palmer 1998)

- ''Helps in clarifying the responsibilities of the team members
- Is able to plan and prioritize work and accomplish planned targets
- Translates requirements into actionable outputs with timelines
- Sets realistic objectives and timeframes
- Reviews team progress against goals
- Ability to be flexible and adaptable in an evolving environment
- Understands human needs, psychology and fears
- Keeps updated on new techniques, theories, methods etc.
- Good communication skills
- Ability to be an effective advocate for the team
- Advocate for needs of internal and external customers
- Ability to lead and to impress the team members''

*Technical Responsibilities*: (Humphrey 2005) (Palmer 1998)

- ''Plans the team's work and coordinate it with the Project Manager
- Converts business objectives into actions
- Monitor and motivate the team's work, builds healthy team climate
- Raise important issues and discuss them with the Project Manager
- Carry out quality control over the team's work

- Ensure that all identified risks are mitigated
- Ensure that the appropriate resources are assigned to the tasks and monitor the effectiveness of the team
- Assure that all team members have the required knowledge and training
- Supports the team in the phase of finding out the: customer needs, specifications, design standards, techniques and tools to support the task performance
- Establish meeting times, places and agendas
- Organizes meetings for coordinating the work progress with the project manager and functional management
- Provide status reports over the team activities against the program schedule''

3.2.2.3 Business Analyst

Requirements play a vital role in the software development process and improper requirements gathering may end with a software development process failure. It is the role of the business analyst that assures that the requirements are captured and fully understood by the technical team before moving to implementing them into solutions. The business analyst is the connection between the business part and the technical providers throughout the software development process. He defines and documents the requirements and this textual representation of the future system is an intermediate step between the software need and the solution design. This design process is divided into business need identification, scope definition and elicitation. (Hass 2005) (Paul, Yeates & Hindle 2006)

The first step is a pre-analysis, which is concerned with detailed research over the business needs; feasibility studies; solution trade-off analysis and development of high level business requirements. Then follows the scope definition, where are included all documents about the description of the initial requirements: the Business Case, Project Charter, or Statement of work. (Hass 2005) And the last step – the requirements elicitation is expressed in the clear description of all stakeholders', customers' and users' needs.

In order to capture the complete and accurate list of requirements the Business analyst must possess a special skill set in the form of the following personal competencies and technical responsibilities.

*Personal Competencies*: (Hass 2005)

- ''Proper communication of technical concepts to non-technical audiences (customers)
- Ability to conceptualize and think creatively
- Time management and personal organization
- Is able to diagnose problems effectively
- Asks appropriate questions to resolve issues and elicit requirements
- Strategic and business thinking
- Effective communication of business concepts to technical audiences
- Understands information quickly and accurately

- Demonstrates clarity in written and verbal communication
- Creates effective presentations to get one’s ideas across
- Problem solving, negotiation and decision making
- Customer relationship management
- Encourages fellow team members to make innovative contributions and embrace new ideas‘’

*Technical Responsibilities*: (Hass 2005) (Hass 2007)

- ‘’Knowledge of system engineering concepts and principles
- Knowledge and efficient application of complex modeling techniques
- Technical domain knowledge
- Fundamentals of project management
- Techniques to plan and document requirements
- Requirements risk assessment and management
- Cost / benefit analysis
- Documents analysis in agreed artifacts and models using standard notation and language understood by business users and other stakeholders
- Business improvement and Reengineering
- Business writing; Business case development and Business domain knowledge
- Proactively tries to understand customers needs and displays commitment towards meeting them
- Elicits and documents business, organizational and operational requirements
- Evaluate customer business needs, thus contributing to the strategic planning of information systems and technology directions
- Identify and understand the business problem and the impact of the proposed solution on the organizational operations
- Document the complex areas of project scope, objectives, added value or benefit expectations, using an integrated set of analysis and modeling techniques
- Liaise with major customers during preliminary installation and testing of new products and services
- Analyze and manage requirements risks
- Conduct root-cause analysis of the problems
- Performs specified data analyses and studies as directed (including research, surveys and feasibility) supporting potential projects
- Measure the value of new business solutions and compare to the estimated benefit‘’

3.2.2.4 Software Architect

The software architect builds the software architecture. He transforms the requirements for the software into an architecture that describes the top-level structure and identifies the software components. His responsibilities are emerging from the conceptualization and experimenting with alternative architectural approaches through developing models and documents to validating everything against the software requirements. (Laporte et al. 2007)

Software Architect should possess the following personal competencies and technical responsibilities:

*Personal Competencies*: (Rozanski & Woods 2005)

- ‘’Makes quick and effective decisions
- Makes decisions from a basis of a holistic understanding
- Identifies problems at an early stage and takes corrective action
- Empowers team members
- Resolves conflicts and interpersonal issues swiftly
- Values continuous development of people
- Leads own team effectively
- Is able to guide or resolve performance related issues
- Negotiates and asserts oneself
- Is able to persuade others to one’s point of view
- Makes effective presentations
- Can synthesize technical and other information to add clarity for others
- Maintains and develops relationships with potential as well as existing customers
- Sets clear checkpoints and targets
- Involves team members when planning and scheduling
- Knows human strengths and limitations and uses this knowledge in planning
- Sets the right priorities when conflicts arise
- Optimizes the use of people and resources
- Is constantly in learning mode
- Facilitates change across own function or team
- Has a habit of seeking improvements in all aspects of one’s work
- Has good listening skills
- Can turn a hostile interaction into a positive outcome’’

*Technical Responsibilities*: (SEI 2011) (Laporte et al. 2007)

- ‘’Is able to analyze the consequences of action
- Represents internally and externally the best interests of the organization
- Produces a very high level of written communication suitable for organization-wide or external consumption
- Has a wide knowledge of industry

- Sees all actions from a dual view-point – of both the organization and the customer
- Continuously reviews status of plans
- Constantly benchmarks own area of operations
- Has a complete understanding of the business domain
- Has knowledge of software life cycle
- Assesses potential projects in relation to current strengths and weaknesses'' (SEI 2011)
- ''Defining the architecture of the software
- Derive the requirements for the software architecture
- Identify the key design issues that must be resolved to support successful development of the software
- Generate one or more alternatives and constraints for the architecture
- Allocate the software and derived requirements to the chosen architecture components and interfaces
- Maintain requirements traceability
- Describe the software architecture by capturing the design results
- Identify appropriate derived requirements that address the effectiveness and cost of the life-cycle phases
- Document, approve, and track the technological changes
- Preparing risk mitigation strategies'' (Laporte et al. 2007)

3.2.2.5 Software Developer

From a technical point of view the developer is at the most basic level in the software hierarchy expected to be able to translate algorithms and technical specifications into functioning software code. He has to have good programming language skills and logical way of thinking in order to transform the specification into particular functions. Of course this knowledge is only the basis for the rest competencies that a good programmer must possess.

*Personal Competencies*: (Klipp 2009)

- ''Brings in fresh perspective (an unbiased point of view)
- Generates new and imaginative ideas/ approaches
- Is flexible in aligning personal objectives with team objectives
- Is a good team player - Is able to work harmoniously as a part of a team
- Tolerates dissent and different viewpoints
- Shares information and data
- Shares opinions and expresses himself confidently
- Asks and answers questions effectively
- Gives his best to understand and meet the customer requirements
- Able of effective time management
- Able to prioritize activities

- Uses failures as a stepping stone to growth
- Is able to articulate own thoughts as well as on behalf of others
- Is open-minded and willing to learn
- Appreciates suggestions and new ideas
- Take full responsibility for own work
- Continuously upgrades to the new technologies, tools and business training
- Display structured thinking and objectivity in analyzing complex tasks''

*Technical Responsibilities*: (Klipp 2009) (Humphrey 2000)

- ''Knowledge and usage of relevant engineering processes
- Knowledge and usage of relevant Standards, Templates, Checklists, Defect Prevention
- In-depth knowledge of relevant software environment and related tools (programming languages, operating systems, databases, debugging & testing tools)
- Configuration Management
- Adequacy of Test Planning - Unit Testing
- Able to understand design specifications
- Approve software only when it works correct, safe, has been tested enough and cannot cause harm to the people or the environment
- Identify, document, and report to the client or the employer possible problems with the software (functionality, cost and so on)
- Fully understand the software specifications
- Good documentation of the software and matching periodically to the users' requirements
- Ensure adequate testing, debugging, and review of software and related documents
- Maintain the integrity of data
- Disclose to all concerned parties the unavoidable conflicts of interest
- Take responsibility for detecting, correcting, and reporting errors in software and the influenced documents
- Integrating software units into the system
- Providing support in the testing of software elements''

3.2.2.6 Software Tester

The job of the Software tester is to perform testing of the application. "Software Testing is a process of verifying and validating that a software application or program meets the business and technical requirements and works as expected". (Bentley, Bank & NC 2004) The software tester works with the Business analyst, the Software Architect and the Developer to convert the requirements and design documents into a set of testing cases and scripts and then to report the occurred problems. These testing cases and scripts can be used to verify that the system meets the client needs.

The software tester is mainly responsible for creating test cases and scripts, executing them and facilitating or performing random testing of all components to ensure that there's not a random bug affecting the system. Here follow his competencies and responsibilities so that he can fulfill his job with the expected accuracy.

*Personal Competencies*: (Perry 2006) (Watkins 2004)

- ‘‘Brings creativity and is open for others’ ideas
- Shows flexibility in approach to task
- Communicates clear and open in order to support the team
- Responds quickly to customer problems
- Is open-minded and willing to learn
- Appreciates new suggestions
- Discusses issues with relevant colleagues to resolve ambiguity’’

*Technical Responsibilities*: (Dustin 2002) (Watkins 2004)

- ‘’Good understanding of GUI design
- Proficient in software testing techniques
- Proficient in the business area of application under test
- Understands various testing phases
- Proficient in working with testing tools’’ (Dustin 2002)
- ‘’Working together with the Quality Assurance part of the team in order to build the test strategy and the test plans
- Defining the test requirements
- Performing the functional analysis of the application under test
- Designing and implementing the test scripts and test cases
- Design, specification and implementation of the test environment and the test data-sets
- Backup and archive-maintenance of the test environment and the test data
- Executing test scripts and observing the test results
- Documenting the test results and maintaining the records
- Identification and recording of any observed faults
- Performing retesting after fixed faults
- Assuring backup and archive of all testing documentation and materials’’ (Watkins 2004)

#### 3.2.2.7 Quality Engineer

According to (Kasse 2004)**,** the Quality Engineer should be able to assure visibility into the projects’ processes for the understanding of the management team and to determine if they are efficient and effective. Also this role is concerned with the necessary product quality, which has to satisfy customer, competitor and organization or project quality goals. The Quality Engineer has to validate the developer’s tests, to ensure that the work of several developers fits together and to follow different standardization methodologies. The main goal of this role is to assure the awaited performance of a software solution.

The following competencies and responsibilities have to be fulfilled for a successful Quality Engineer.

*Personal Competencies*: (Daughtrey 2001) (Kasse 2004)

- ‘’New ideas, flexible personality
- Contributes to the team objectives
- Remains positive at all times and focused on opportunities
- Systematic and organized personality
- Tolerates others opinions and defends his own
- Able of effective time management, resource estimation and allocation of decisions
- Appreciates suggestions, new ideas and others’ opinion
- Seeking for knowledge
- Gains co-operation from others
- Ability to understand customer business
- Ability to convince the customer
- Ability to conceptualize and provide right solution
- Ability to present data in an effective manner for decision making
- Ability to interact with senior managers, especially for developing the quality goals
- Ability to track progress according to plan and deviations from it
- Ability to interact with external consultants and vendors’’

*Technical Responsibilities*: (Kasse 2004)

- ‘’Setting & monitoring Quality goals and metrics.
- Review of milestone analysis, closure reports.
- Knowledge of methodologies and software engineering concepts.
- Ability to conduct project reviews and audits.
- Has a good understanding of SWOT (**S**trengths, **W**eaknesses, **O**pportunities, and **T**hreats) analysis in terms of own business area
- Strong understanding of S/W Engineering processes and selecting an adequate set of standards, practices, and procedures
- Basic knowledge & usage of S/W Metrics
- Knowledge & Usage of software life cycle tools software project management tools and its applicability
- Knowledge & Usage of statistical techniques
- Ability to identify problems, data gathering, generating alternatives, root-cause identification, objective analysis of data, generating solutions
- Resource planning
- Ability to track utilization of budgets
- Ability to plan large/cross functional initiatives

- Negotiating criticality levels for the product components and subsystems
- Performing ad-hoc process compliance evaluations
- Co-working with the appropriate customer representatives on process and/or quality problems
- Approving the supplier's quality plan and resulting implementation
- Assess the projects and organization's Configuration Management activities to ensure the integrity and consistency of the work products
- Assuring adequate testing of the software components that fits the development process''

## 3.3 Summary over the Software Engineering and the Software Roles

In the first part of the chapter were explained the basics of the software engineering, which could be shortly summarized in the already mentioned formula:

$$SE = (M_{SE}, R_{SE}) = (\{SE\text{-}Methods, CASE, SE\text{-}SystemOfMeasures, SE\text{-}Standards, SE\text{-}SoftwareSystems, SE\text{-}Experience, SE\text{-}Communities\}, R_{SE})$$

The overview about the high quality software product, as the main point of the software engineering is expressed in the following: a software system/product *SP* is developed by the software process/development *SD* and is based on the supporting resources *SR*. As we have already seen one of the major resources is the software personnel.

$$SR = (M_{SR}, R_{SR}) = (\{personnelResources, softwareResources, platformResources\}, R_{SR})$$

Explanations about the Software Development Process, the Use of the Software Product and the Software Maintenance make the overview of the software engineering complete and comprehensive. One of the major points is the Software Project Risks as part of the software development, where we could clearly see the big number of risks connected with the personnel. For example:

- Lack of experience and specific knowledge
- A lot of outsourcing
- Lack of understanding of the business-processes
- Not flexible organization structure
- Lack of goal validation

The different involvements of human risks in the software engineering area are summarized as following.

Considering software risk-based processes

$$r_{SE}^{(RiskManagement)} \in R_{SE}: \textit{SE-RiskAssessment} \times \textit{SE-RiskControl} \rightarrow RiskManagement \quad (3.46)$$

$$r_{SE}^{(RiskCommunity)} \in R_{SE}: \textit{SE-Communities} \times systemOfRiskMeasures \times RiskManagement \rightarrow RiskMeasurementStaff$$

$$r_{SP}^{(riskIdentification)} \in R_{SP}: dataRisks \times applicationAnalysis \rightarrow riskIdentification$$

$$r_{SP}^{(riskManagement)} \in R_{SP}: riskIdentification \times riskAnalysis \times riskPrioritization \times riskMgmtPlanning \times riskResolution \times riskMonitoring \rightarrow riskManagement$$

$$r_{SD}^{(waterfallRisks)} \in R_{SD}: problemDefinition \times specification \times design \times implementation \times acceptanceTest \times riskManagement \rightarrow waterfallModelRisk$$

$$r_{SD}^{(VmodelRisks)} \in R_{SD}: (problemDefinition, softwareApplication, riskManagement) \times (specification, acceptanceTest, riskManagement) \times (design, integrationTest, riskManagement) \times (coding, unitTest, riskManagement) \rightarrow VmodelRisk$$

$$r_{SD}^{(processRisk)} \in R_{SD} = businessFocus \times organizationStability \times marketDynamic \times systemCriticality \times specialRisk \rightarrow processRisks$$

$$r_{SR}^{(personnelRisks)} \in R_{SR}: HF_{IT} \times processInvolvement \times role_{IT} \rightarrow personnelRisks$$

$$r_{SR}^{(personalAssessment)} \in R_{SR}: person_{IT} \times assessment \times workingProcess \rightarrow personalAssessment$$

$$r_{SR}^{(supervisorAssessment)} \in R_{SR}: person_{IT} \times supervisor \times assessment \times workingProcess \rightarrow supervisorAssessment$$

$$r_{SR}^{(colleagueAssessment)} \in R_{SR}: person_{IT} \times colleague \times assessment \times workingProcess \rightarrow colleagueAssessment$$

$$r_{SA}^{(applicationRisk)} \in R_{SA}: deliveryRisk \times migrationRisk \times operationRisk \times outsourcingRisk \rightarrow applicationRisk$$

$$r_{SM}^{(maintenanceRisk)} \in R_{SM}: extensionRisk \times correctionRisk \times adaptationRisk \times improvementRisk \times preventionRisk \times remoteMaintRisk \rightarrow maintenanceRisk$$

and considering software process risks aspects

$$dataRisks = \{missing, incorrect, incomplete, not\ synchronized, misleading\} \quad (3.47)$$

$$requirementsRisks = \{incomplete, unrealistic, subjective, dependability, dynamic, incompatible, not\ measurable\}$$

$$personnelRisks = HRF_{IT}$$

$$softwareResourcesRisks = \{notAvailability, highCosts, incomplete, incompatible, veryComplex, difficultyByChanges\}$$

$$hardwareResourceslRisks = \{lowPerformance, deadlocks, highCosts, incompatibility\}$$

$$humanPerformance = \{HF_{IT}, softwareDevelopmentProcess\}$$

$$humanPerformanceEvaluation = \{personalAssessment, supervisorAssessment, colleagueAssessment\}$$

$$risksInUse \subseteq \{lackOfExperience, lackOfResources, strongDependencies, lackOfUnderstanding, notFlexibleOrganization, lackOfGoalValidation, highSystemComplexity, badInformationStructure, lackOfData\}$$

Having explained the basics of the software engineering we have moved forward to the software team and we have observed the seven basic roles that are met in every kind of software company.

First we have made a research about the possibilities for organizational structure in order to find the most common one – the matrix organization and then we have observed in detail the roles and their responsibilities inside.

The general characterization of the considered personnel resources is defined as following

$$personnelResources = person_{IT} \cup person_{customer} \cup person_{applicatiuon} \quad (3.48)$$

$$person_{IT} = \{analyst, designer, developer, acquisitor, reviewer, programmer, tester, administrator, qualityEngineer, projectLeader, systemProgrammer, chiefProgrammer\}$$

$$person_{customer} = \{stakeholder, manager, acquisitor\}$$

$$person_{application} = \{user, operator, client, consumer\}$$

Therefore, we will summarize the chosen personnel – as ***seven basic roles of person$_{IT}$*** - and their competencies like the following:

$$HF_{ProjectManager} = \{\textit{communicative, managerial skills, disciplined, respects the others, resolves conflicts, open minded, willing to develop himself, well-organized, goal-oriented, seeks improvement}\} \quad (3.49)$$

$$HF_{TeamLeader} = \{\textit{plan and prioritize the work, reviews team progress, flexible and adaptable, communicative, an effective advocate for the team, ability to lead and to impress}\}$$

$$HF_{BusinessAnalyst} = \{\textit{communicative, conceptual thinking, creativity, strategic and business thinking, problem solving, negotiation and decision making, customer oriented, team player}\}$$

$$HF_{SoftwareArchitect} = \{\textit{good decision maker, team player, performance oriented, technical understanding that supports the team, optimizing abilities, seeks new knowledge}\}$$

$$HF_{SoftwareDeveloper} = \{\textit{creativity, team player, tolerant, always in a learning mode, able to articulate own thoughts, respects others' ideas, structured thinking}\}$$

$$HF_{SoftwareTester} = \{\textit{creativity, flexibility, communicative, open-minded, respects the others}\}$$

$$HF_{QualityEngineer} = \{\textit{flexible, team oriented, positive attitude, systematic and organized, respects the others, seeking for knowledge, convincing ability, ability to interact with managers and customers}\}$$

These competencies will be used in the following FMEA analysis in the next chapter in order to discover the human factors that influence at most the software engineering process and the corresponding failure modes.

# 4 Chapter – Discovery of the IT Human Factors

Based on the specific personal competencies discovered in the previous chapter, here the goal is to analyze the responsibilities of each IT roles in order to find where the weak places could be. We are using a well-accepted method for failure analysis – the FMEA as it gives the possibility to analyze each process, to find the weak points and the influencing factors behind. These influencing factors are actually the IT human characteristics which we will evaluate in the next chapter 5 in order to find the personal productivity in the software development process.

## 4.1Classical Failure Mode and Effect Analysis

Progress - this is the heart of the Failure Mode and Effect Analysis (FMEA). The constant need of change and improvement is the engine, keeping the FMEA process running. This idea may not be new, but is done in systematic way to address problems and failures and to search solutions for progress.

FMEA is defined as a specific methodology for estimation of system, design, process or service for possibilities of occurring of failures like errors, risks and different concerns (Stamatis 1995). When a failure is found, it is evaluated with occurrence, severity and detection characteristic. So depending on the values of these marks, an action is taken, planed or ignored. The idea is to decrease the likelihood of a problem or its consequences.

The main goal of FMEA is to predict the problems before they occur, to make the product safer or optimize the process and to lead the company during the production process in order to satisfy the customers' needs. Usually, there are two main kinds of FMEA – over an existing product – Product FMEA and over process development stages – Process FMEA. When product and process FMEA are conducted together they significantly reduce the costs of manufacturing and developing. It is considered that process FMEA is more important because of the early stages where the failures can be detected and prevented which gives a result of more robust process and no need of post-the-fact corrective actions.

Nowadays, FMEA is part of every quality system, which means that collecting the right information and making conclusion is not the only part. In order to get the maximum, the company needs to implement the proposed improvements that are the results of the FMEA. The reasons of conducting an FMEA and the benefits are proven and more than clear: (Stamatis 1995)

- ''Improved quality, reliability, safety of the products or services.
- Improves the company's image and competitiveness.
- Increased customer satisfaction.
- Reduced product development time and costs.
- Helps determine the redundancy of the system.
- Helps define the corrective actions.
- Helps in identifying errors and their prevention.
- Helps decide the priority of the failures and associates the right preventive operations.

- Helps reduce the customers complains.
- Increases the productivity.
- Develops early criteria for development.''

### 4.1.1 Concept of the Failure Mode and Effect Analysis

After all FMEA is an engineering method used for the first time in aircraft building and car manufacturing so it is described as part of some industries, like a quality standard. When a particular organization succeeds in implementing these standards it is capable to control the processes and determine acceptability of its products or services.

Every FMEA method performed in the right way provides the company with useful information which can be used efficiently – to reduce the work, optimize the processes or prevent serious loss. Due to the consecutive and constructive method the task can be performed more effectively. The early study of possible problems is with significant importance and every failure is evaluated for its effects on the whole system, product or process.

If the method is used in a corrective way it shows the actions to prevent failures, reaching the customer and raises the reliability and quality of the process or product. The process of conducting FMEA looks like shown on the next figure, where we can see four main steps, which we will discuss later on. (Stamatis 1995)

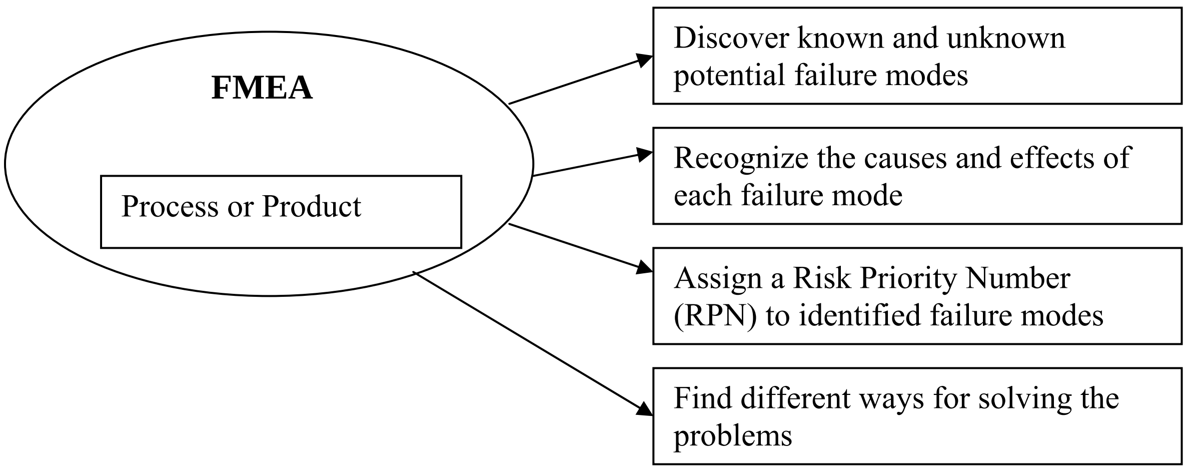



**Figure 23 The FMEA Process (Stamatis 1995)**

In another aspect FMEA is a method to bring satisfaction of the customers. In modern world we know that the most important thing in order to stay on the market is having qualitative products. The main key here is to achieve detection of quality concerns before the product reaches the hands of the users. That is why FMEA should start as soon as some information is provided, because the team conducting the FMEA will practically never have all the data. At the beginning, the technique should be executed over the design stage or concept, but for better results it can be used throughout the development process and the whole product life cycle to identify failures. Every product is expected to do something specific and to be in use for long time. A product failure is when it does not function the way it is expected to. Even the simplest products can malfunction in some way.

FMEA includes everything that can be done in order to make the product work closer to 100 percent – this means even the problems that occur during the exploitation of the product.
In the cases when the product malfunctions or fails to work we talk about failure modes. Each failure mode should be described – with what frequency it occurs and what damages it's leading to, how the system is affected.

#### 4.1.1.1 Types of FMEA

*System FMEA* (Stamatis 1995):

It is applied over systems and their interaction. Its focus is the function failures in the system. The benefits that it brings are:

1. Identification of system alternatives
2. Discover redundancy
3. Potential for managing future problems
4. Recognition of failures in the systems' interaction

*Design FMEA* (Stamatis 1995):

Used for analysis of products ready for manufacturing.
The profit from the design FMEA:

1. Prioritizing the design improvement actions
2. Information for product design validation and testing
3. Defines alternatives for design requirements
4. Mitigation of safety issues

*Process FMEA* (Stamatis 1995):

This is performed when manufacturing and assembly process is being analyzed.
The advantages from this FMEA are:

1. Recognition of the process deficiencies
2. Proposing and prioritizing the corrective actions
3. Exposure of the manufacturing or assembly process
4. Track down the meaningful changes

*Service FMEA* (Stamatis 1995):

This FMEA is analyzing services, before they come to the customer.
The gain from the performed service FMEA can be observed in the facts:

1. Helps to evaluate the job flow
2. Exposure of the system and process
3. Implementation of a control plan
4. Prioritizing improvement actions

### 4.1.2 The methodological steps in the FMEA

In order to achieve problem solving results, FMEA needs to be conducted strictly, consecutively and constructively, following 8 main steps: (McDermott, Mikulak & Beauregard 2009), (Stamatis 1995)

*Step 1*. Gather a team and review the process or product.

*Step 2*. Brainstorm unknown risks.

*Step 3*. Assign different effects caused by the failures.

*Step 4*. Prioritize – assign severity, occurrence and detection rankings for each failure mode.

*Step 5*. Calculate the RPN number.

*Step 6*. Collect data, analyze and measure the failure modes for action.

*Step 7*. Apply methods to reduce high-priority/high-risk failures.

*Step 8*. After performing actions evaluate the performance of the system again.

The Bottom-up approach of FMEA looks like shown on the figure:

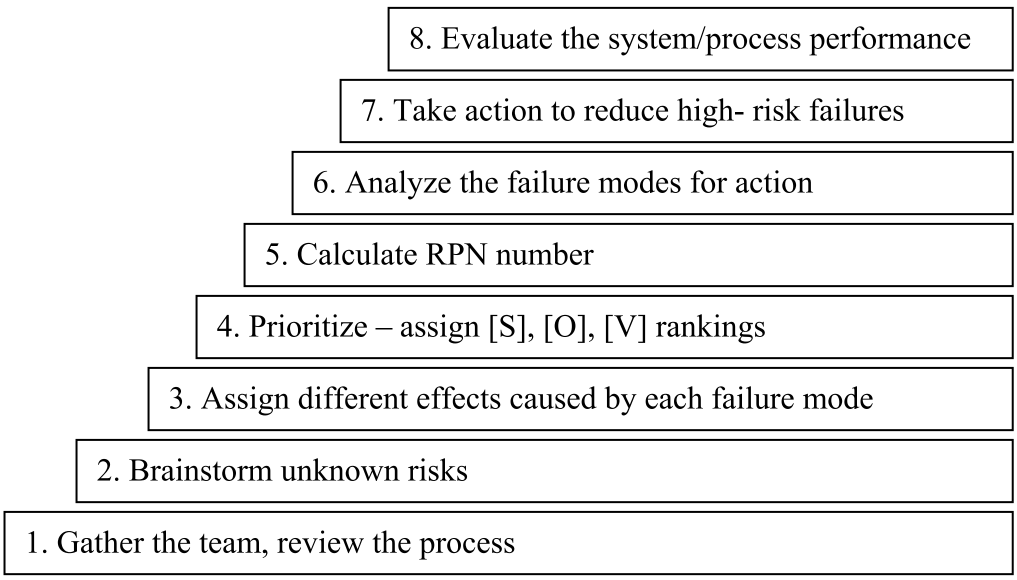



**Figure 24 The bottom-up approach of FMEA**

*Step 1: Gather the team, review the process*

When gathering the team we must know that the proper people are going to take part. Everyone should know the field of the work and prior to the start of the FMEA the team leader has to make available for everybody a detailed flowchart of the development process if they are conducting a *process FMEA* or engineering drawing of the product in case of *product FMEA.* Sometimes it is recommended to have an expert in the group available for answering questions and giving useful hints.

*Step 2: Brainstorm unknown risks*

Having a good overview over the process or product, the team is ready for brainstorming. The members try to brainstorm any kind of ideas and various suggestions about what could affect and impact the process or the product quality and stability.
Because of the big variety of topics it is recommended to be conducted several brainstorming series and every one of them should focus on different elements of the *Process FMEA* – people, resources, equipment, methods. This, of course, helps for deeper understanding and finding of failure modes.

*Step 3: Assign different effects caused by each failure mode*

In the computer programming – we can explain this step as: if {} then {} construction. The team should think: *If* a problem occurs, *then* what the consequences are. In some cases failures can cause several effects, but in other – only one. This step is very important because of the further assigning of Severity and Occurrence.

*Step 4: Prioritize – assign Severity, Occurrence and Detection rankings*

After examining every risk carefully the team puts every effect in a table, describes the influence and assigns a rank from 1 to 10 for every of the three (severity, occurrence and detection) components. Every member should be able to understand the rankings – the more descriptive explanations for every ranking scale – the better the FMEA process is.

*Step 5: Calculate RPN number*

This number most of the times serves as a guide and is not taken under serious importance because of the different effects of every failure mode.
However, it can be used as an instrument for measuring – if it's under defined value the team does not take any action. Calculation:

$$\text{Risk Priority Number} = \text{Severity} \times \text{Occurrence} \times \text{Detection} \tag{4.1}$$

*Step 6: Analyze the failure mode for action*

In this step every of the failure modes is analyzed by ranking and the effects and is given a priority for action. The team decides which the highest risks are and where to put work on.

*Step 7: Take action to reduce high-risk failures*

Probably this is one of the most important steps where the team decides what actions to implement in order to reduce as much as possible the severe problems. The ideal case is when no future failure modes are observed but it is not always achievable. At least the team must aim in increasing the detection and mitigation of the failure.

*Step 8: Evaluate the system/process performance*

After implementation of the methods for reducing failures the team continues to measure the performance of the process/system, confirms the results and performs another FMEA.
Recommendations should be made after answering the questions:

- Is the process better than before?
- Are the improvements enough to have good RPNs?
- Is it urgent to conduct another FMEA?

Every organization, according to its resources and budget takes own decision how many FMEA analyses should conduct. Nevertheless, the long-time goal is always to eliminate every risk and the short-time goal is to reduce the impact as much as possible.
After all, we have to remember – *FMEA* is a *continual method of improvement*.

4.1.2.1 The FMEA Parameters

The project team analyzes every element of the process, working through the entire output which has to be delivered to the customer. In every step the team tries to brainstorm and find unknown and potential problems and offer solutions to already known risks. Every problem is estimated and has different priority. It is very important to have a scale of measuring so the team knows which risks are critical for the system.

There are tree indicators the team uses to define the priority of the failures:

- Severity - [S]
- Occurrence - [O]
- Detection - [D]

***Severity*** **-** shows what the impact of the failure is over the system or over processes, how serious the consequences are. After all, the main goal of FMEA team is to take actions and reduce the most important failures. The team uses a scale from 1 to 10 to express how serious one problem can be – as 1 stands for ‘no danger’ and 10 for ‘critical’. These numbers help to prioritize the risk and help in focusing on the serious risks. Examples of failures are:

malfunction in UPS system which leads to data loss, or improper use of variable in accounting software, which results in loss of accuracy. Another important reason why we use this rank is that – we may face a failure which leads to another failure or component disability. (Stamatis 1995)

***Occurrence* -** this measure shows us how often a failure occurs. The team has to have in mind also the severity number at this step. Examples: how often we come into program failures because of an erroneous algorithm or how often hardware experiences excessive voltage. With essential importance here is that the team must find the cause of the failure. Again we use a ranking – occurrence ranking [O], from 1 to 10. If the rank is high (above 7), precocious mechanisms should be determined. But sometimes in a situation where occurrence is not high but the severity for the failure is with rank above 8 the team must also react. On this step it is always necessary to look for the severity rank with combination of occurrence. (Stamatis 1995)

***Detection* -** the chance or the capability of the team to detect the failure before it reaches the customer. The last two steps work in combination and every combination of them is marked with a detection number which shows what the possibility that the failure will not be detected is. A high number of Detections means a higher chance that the failure will escape detection. (Stamatis 1995)

***RPN Number* -** when the last three steps are completed, an *RPN* (Risk Priority Number) has to be calculated. It shows us which of the process steps and parts are under high risk and have to be taken under control measures. The number is calculated by multiplying the Severity, Occurrence and Detection numbers:

$$RPN = S \times O \times D \qquad (4.2)$$

RPN numbers are calculated for every system/process and every sub-system/sub-process in order to find where the critical parts are. The sub-process with highest RPN number needs a corrective method to be applied and it is not always the severity numbers the ones which define this, for instance, it could be failure which is hardly detectable and occurs quite often, but does not have serious effect. (Stamatis 1995)

All the steps and entire FMEA process should be documented using a worksheet. There are different kinds of worksheets according to the types of FMEA. The form captures all the information in a clear and well-organized way. Everything is included – recommended measures and methods, implementations and all the numbers for Occurrence, Severity, Detection and RPN. Once the team has all the information they have to face with four main objectives:

- ✓ Reduce the impact of the failure mode.
- ✓ Minimize the severe effect as much as possible.
- ✓ Try to eliminate the occurrence or put the levels as low as it can.
- ✓ Improve the occurrence detection.

### 4.1.3 Software FMEA

Technical systems are used in a big variety of areas in the worldwide industries. Considerable amount of software specialists and software code is used to move forward these industries. As a consequence, fairly large attention is focused on the identification and avoidance of technical risks and failures. A very powerful tool for analysis, preventing and predicting errors is the systematic and constructive method – FMEA, which is approved and accepted in many different fields of manufacturing – cars, airplanes, computers… In most of the cases a bottom-up technique is used to identify failures and malfunctions in every component of the system or process. (Mäckel 2006)

First the method has been used in the military, but the concept and the ideas put there are not compatible and do not apply in the modern technologies and therefore, the companies nowadays have developed new sets of priorities, guidelines, rules and standards of their own use. FMEA based on hardware and system levels is well understood, applicable and working in a good way because of the known risks and failures of the hardware behavior. But in present times the accent is on the software level – more systems and functionalities are based on the software process, which explains the need of software based FMEA.

"Software modules do not fail, they only display incorrect behavior" (Pentti & Atte 2002)

Anyway our goal is not to focus over the SFMEA but only to show that it has its application in the software industry and in this way to motivate our modification of the method over the human actions during the software engineering process.

SFMEA is also a step by step systematic method for analyzing the software architecture, software design or process with taking care for the technical risks – reliability, safety, stability, availability and so on. Big advantage of the method is to use the information and documentation from every department which is taking part in the process of development – system, software, test and service, so the FMEA team is able to have a clear and a deeper insight of the problems.

The figure below shows which the critical moments in the software life cycle are and where the FMEA should take part: (Mäckel 2006)

There is relatively little information published on the use of FMEA for software systems but we will provide a short overview of the papers discussing the benefits introduced by the SFMEA. Banerjee has applied the method in the practice and observed an ''improvement of the reliability of the software production process, resulting in higher product quality as well as in higher productivity''. (Banerjee 1995)

The statement that detailed SFMEA validates that the software has been planned and constructed to reach the right and safe requirements from the beginning is being defended also in the following scientific work: (Pentti & Atte 2002), (Lauritsen & Stalhane 2005), (Hartkopf 2004), (Ozarin & Siracusa 2003), (Bowles & Hanczaryk 2008), (Nguyen 2001) and (Goddard, Raytheon & Troy 2000). The authors point out that the use of the FMEA in the software process brings early identification of potential software failure modes and is an excellent practice that supports the whole life-cycle, in the same time each of them has demonstrated a concrete application of the FMEA method in the software development process.

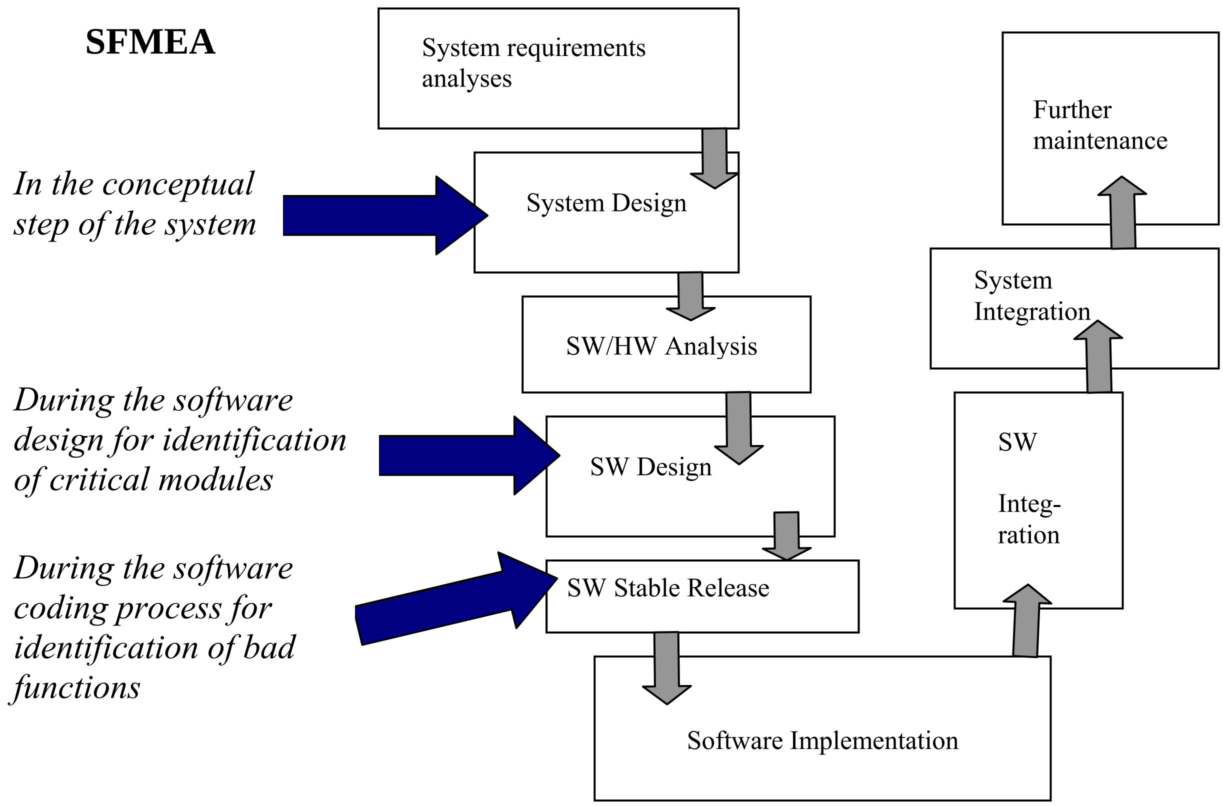



**Figure 25 Application of the Software FMEA, (Mäckel 2006)**

Motivated from this wide use of the FMEA method, due to its universal manner we have decided to apply it over the software development phases but pointed to the human roles and the actions that they perform.

## 4.2 Adopted FMEA for the Software Personnel

We have already introduced the FMEA methodology. Its evaluation and failure detection were broadly explained. Considering the FMEA strong points in analysis and corrective recommendations, the decision of applying it first over the software development process (Georgieva, 2010 d) and then over the software development roles and their responsibilities was logical and consecutive (Georgieva et al, 2011 a).

The price of the human errors that we all pay in the everyday life, facing the software applications can be very high. Therefore detecting and decreasing their effect is a vital development step in each production process or system. With a Software Human Factor FMEA could be made an evaluation over the human failure modes severity and occurrence and in this way these errors can be ranked according to their criticality. The other analysis that can be done adopting the FMEA technique is to discover the human features that stay behind these errors, problems or failures. Nevertheless our goal is not only to find the possible risks but to find out why do they occur and to try to resolve them. On this step the chosen method gives us a special benefit, as it delivers the information why a particular failure mode appears

and what is the employee's fault, what could be his personal characteristics that led to this problem.
We will observe in the application of the adopted method the discovery of the human factors, or the specific human features that stay behind the different failure modes in a particular software team.

### 4.2.1 Performing the Software Human Factor FMEA

The form in which the FMEA analysis is performed could be changed in every company and could be adapted to the particular goals and expected problems. We will show first the possible entries in such analysis and then will concentrate on the chosen fields that are important in our case. We will focus on Process FMEA as the activities that we want to analyze are actually the different software phases and the human actions inside, which is nothing else but a number of processes.

In the form, presented in Table 5, are listed the generally expected entries that should be managed when conducting a process FMEA. For our research we have adopted the method and added the column Human Factors that will give us the essential information for the further research over the criticality of the personal features in the software process.

The first part between 1 and 9 is the introduction data. These are not mandatory fields, however, they bring information that may be important in future examination. The main part are numbers 10 to 23 – these are mandatory items and are the essential part of the FMEA conduction. Additional to the form there are signatures, which may not be mandatory, but bring an authority look to it and can be a sign that the analysis is ready. Here are presented the 23 items according to Stamatis. (Stamatis 1995)

1. *Process Identification*: Here is stated the name of the process or a reference number, adding identity to the process that is manipulated.
2. *Manufacturing or design responsibility*: The prime responsibility is stated here: this may be the name of the activity, machine or material.
3. *Involvement of other areas:* Mention if other people or systems are connected to this part.
4. *Involvement of suppliers or other:* When additional persons are taking part in the design, manufacturing or assembly of the part.
5. *Model or Product:* This is the place to be specified the name of the model or product using the process.
6. *Engineering Release Date:* The planned date for release.
7. *Prepared By:* The FMEA analyst is stated here as well as some additional information as address, telephone or email.
8. *FMEA Date – Original:* The starting date of the process.
9. *FMEA Dare – Revision:* The date of the last revision.
10. *Process Function:* ''This is the process intent, purpose, goal or objective.''

11. *Potential Failure Mode:* This is the possible problem, failure or defect. This is where the person can go wrong. Each action provides the possibility for misunderstanding, omitting, incompletion or falsely interpreting. Therefore, each Process Function may have several failure modes. Each one must be recorded for the future analysis. The potential error should be stated short but clear, this way facilitating the evaluation of the consequences.
12. *Potential Effect of the Failure:* This field is for the result of the wrongly fulfilled responsibilities. Potential problems must be foreseen and tracked down so their effect can be estimated and their occurrence removed. Here again can be written more than one entry. The impact can be observed from several sides, including the influence over the next part of the process and over other related parts of the development.
13. *Severity:* This is a value, assigned for the importance of the effect of the failure. The values are in the area from 1 to 10, where 1 indicates that there is no effect on the process and 10 points catastrophic influence. The exact effect of the failure should be indicated, so that appropriate ranking can be performed.
14. *Potential Mechanism, Causes of Failure:* These are the reasons that cause the already described failure. Here the root cause of the failure must be identified. This is a key item in the analysis, because it directly exposes the human factors that stay behind the potential problem.
15. *Human Factors:* Here are listed the human factors that have most significant impact over the failure, when a team member performs his responsibilities. There are cases, where more than one factor affects the situation.

**Table 5 Software Human Factor FMEA template form**

| | | | | | | |
|---|---|---|---|---|---|---|
| **(1)Process name:** | | **(4)Supplier Involvement:** | | **(7)Prepared by:** | | |
| **(2)Part name:** | | **(5)Model/Product:** | | **(8)FMEA date:** | | |
| **(3)Involvement of others:** | | **(6)Engineering release date:** | | **(9)FMEA rev. Date:** | | |
| | | | | | | |
| **Process Function (10)** | **Potential Failure Mode (11)** | **Potential Effect(s) of Failure (12)** | **Severity (13)** | **Potential Cause(s) / Mechanism(s) of failure (14)** | **Human Factors (15)** | **Occurrence (16)** |
| | | | | | | |

| **Detection method (17)** | **Detection (18)** | **R P N (19)** | **Recommended Action(s) (20)** | **Responsibility &Target Completion Date (21)** | **Action Results (23)** | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | **Action(s) taken and Completion Date (22)** | **Severity** | **Occurrence** | **Detection** | **RPN** |

16. *Occurrence:* This is a numeric value, indicating the frequency with which a failure happens. Again the scale is from 1 to 10, where 10 is constant occurrence. This element is important because it affects the entire priority value of the problem, when calculating the RPN.

17. *Detection Methods:* These are the tools, used to recognize the failure. For the human errors this could be brainstorming, sample filling, daily reports, team meeting or manager's observation.

18. *Detection:* This value shows the rate of the detection of the particular failure. This rating is in the range from 1 to 10 and 10 means every time observing the problem. It must be noticed that this detection is for the likelihood of the error happening to be noticed and not for the particular Human Error.

19. *RPN (Risk Priority Number):* This is the product of the severity, occurrence and detection. It is mostly used to priorities the failures. The RPN has no other meaning apart from the ranking.

20. *Recommended Actions:* Here are listed the activities, that should be taken, so that the failure is mitigated. In the HF FMEA the main object of observation is the Human Factor. So these prescribed actions are mainly intended to correct the reasons of the failure behavior.

21. *Responsible Area or Person and Target Completion Date:* Here must be entered the person, responsible for the recommended actions, and the planned date, on which they should be finished.

22. *Action Taken and Completion Date:* This is one of the actions, filled in the recommended actions list. It is desirable this to be a top activity in the list, guaranteeing maximum increasing of the human performance.

23. *Action Results:* After the recommenced corrective action is done, again a severity, occurrence and detection value is calculated, determining a RPN grade. The new RPN should be better, indicating progress in the person's performance and recovery from the failure.

These 23 steps represent the adopted Software Human Factor FMEA method, used for investigating the human factors behind the employees' performance. For our further investigation are important only the human factors that stay behind the different failures or potential problems and because of this we have taken only a part of the FMEA form, which you will see on the following tables.

We have conducted the FMEA in strictly analytical way over the responsibilities of the software development team members, stated in the previous part. A logical consecutive analysis is conducted in order to define the human features responsible for the variety of mistakes. We have left the RPN and its components out of the analysis as our goal is to find all human factors and not to evaluate them at this step.

4.2.1.1 Software Human Factors FMEA over the Project Manager role

We have analyzed the responsibilities of the Project Manager, which were already listed above. They are just slightly combined so that we have optimized the FMEA table. After having the FMEA result we have built a table with all human factors that influence the performance of the Project Manager.

**Table 6 Human Factors for the PM, extracted from the SHF-FMEA table on the next page**

| **Human Factors for the Project Manager** |
|---|
| Coordination |
| Self-management |
| Overload=Stress |
| Competence |
| Knowledge |
| Effectiveness |
| Concentration |
| Communication |
| Self-Development |
| Liberalism |
| Control delegation |
| Selfish=Egoism |
| Over self-confident |
| Self-organization |

We can express all needed personal characteristics for the Project Manager in the following manner:

$$HF_{ProjectManager}^{FMEA} = \{Coordination,\ Self\text{-}management,\ Stress,\ Competence,\ Knowledge,\ Effectiveness,\ Concentration,\ Communication,\ Self\text{-}development,\ Liberalism,\ Control,\ Egoism,\ Confidence,\ Organization\} \quad (4.3)$$

**Table 7 SHF-FMEA over the Project Manager Role**

| Process Function | Potential Failure Mode | Potential Effect(s) of Failure | Potential Cause(s) / Mechanism(s) of failure | Human Factors | Detection Method(s) | Recommended Action(s) |
|---|---|---|---|---|---|---|
| Ensures that the software development process works as intended | PM is not monitoring the development process closely enough | the development process may be running with hidden problems | developers may issue problems that they do not report to PM and cope with them on their own, providing not so good solutions | coordination, management | Reports, meetings and cooperation between the employees | PM meets regularly the team members and discuss the progress with all accompanying problems |
| Responsible for leading the work process until the completion of the project | PM is not taking the responsibility for the project pointing finger to Developers and team leader (TL) | Developers and TL are being distracted by new responsibilities or changes in the project | PM is not managing the project in the right manner; PM is not aware of his particular duties; | overload, competence, knowledge, effectiveness | Developers not spending all their time coding but rather organizing meetings; scheduling phases and planning events | PM being aware of his duties, so Developers can focus on developing code |
| Coordination btw the development team and the business stakeholders to ensure matching of goals and products in the expected time | PM is not communicating enough with the development team and clients and not monitoring the development process | team may run over the deadline, due to too little control and motivation | PM is looking in future deals and neglects the current ones; PM has left all obligations to development team and believes they will manage alone | competence, concentration, coordination, management | a deadline is crossed; project is not fitting all clients expectations | PM should be constantly tracking the development process asking questions and making himself sure everything is as it should be |
| Working with the business stakeholders, who work closely with the functional analyst during the first stages | PM is omitting meetings with the business stakeholders and FA in the begging | PM is losing time at a later phase for not being clear what is the project about and what has to be done | PM is busy performing other duties; PM believes that early talks do not concern him | overload, competence, coordination | PM is not familiar with solutions specifications and time is needed so he can embrace them and direct them to TL | PM should be more disciplined, always attend pre development meetings |
| Responsible for status reports, that show urgency and demand concrete answers | PM is not insisting on getting constant status report | PM is not pointing urgency and action and is leaving the team to mange alone | PM believes these are TL responsibilities; PM is busy with other tasks | coordination, overload, communication | development team is not hardworking; this may lead to not finished project on time and not fulfilled requirements | PM should be more skilled by team motivation; PM should monitor and report the status in order to use this info on a later stage |
| PM coordinates several projects | PM is working on too many projects | PM mixes the people in different teams, the tasks and the schedules | Too many projects confuse the PM; he is not examining the record with the team and their responsibilities | overload, competence, coordination | PM may come unprepared to a meeting due to a mistake about the team and project | PM should prepare well the documents over the development assigned to the employees |

| | | | | | | |
|---|---|---|---|---|---|---|
| Organizing the team responsibilities | PM is pushing the people to their limit | team is always being pushed without sense of relief or acknowledgement | PM is demanding quick decisions and in this way pushing the team to work on their limit | development, communication, liberalism | team is stressed, always being pushed on | PM should understand the team effort for solving problems; PM should build a good theme atmosphere |
| Executing and controlling the work | PM is not regularly keeping an eye over specific issues | PM may lose track of a problem | PM has a lot of details to keep constant attention to; PM is not responsible for the small issues | competence, control delegation, coordination | In case of well-known problem the PM cannot profit from previous decisions and knowledge | PM should be more focused on the big problems that arise and not on each issue |
| Look at the big picture to evaluate risk, time and costs | PM is not keeping an eye over the development progress | PM cannot evaluate whether the project will be completed on time | PM is delegating this responsibility to TL; PM is having too many projects | coordination, overload | Comparing the current progress with other projects | PM should take care of the project; estimating its properties and details |
| Communicating with the team, to ensure that all problems are correctly understood | at meetings PM is always speaking over a topic, without listening to make sure the members have understood it | PM is overtaking meetings without letting anyone else to say a word and not ensuring that the team has understood him | PM is feeling like a centre figure and does not want to give the word to someone else; PM thinks everyone understands him | selfish, over-self confident, communication | problems may remain unclear; further meetings may be needed; PM may be not understood | PM should listen to his team members and assure himself that everything is clear |
| Makes sure that the process is going according to the requirements | PM is not tracking closely the project | PM is not controlling the project and it can slip away | PM is having too many obligations and is delegating obligations to the TL, who is not fulfilling them | overload, coordination | predicted results may not match the actual results | PM, even if delegating some task to TL must keep an eye on it |
| Document, obtain approval, and track all changes in project parameters | PM is gathering all project details but not documenting them for current projects reference | there is no concrete record for the current project | PM is gathering documents but not organizing them in useful matter | organization, skills, competence | if someone is looking for details over a completed project | PM should be well organized working with tools supporting the good documentation |
| Finalizing the project | PM is not fulfilling the guidelines and cannot finish the project as expected | other team members are considered responsible | PM is believing it is not his task to determine all rules and to give clear 'orders' | coordination, competence | TL coping with too many issues; rather than concentrating on their major specific ones | PM should be aware what are his duties and should strictly perform them |

4.2.1.2 Software Human Factors FMEA over the Team Leader role

Analogically to the Project Manager we have analyzed the responsibilities of the Team Leader, already listed above. They are just slightly combined for optimization of the FMEA table. After having the FMEA result we have built a table with all human factors that influence the performance of the Team Leader, shown below.

**Table 8 Human Factors for the TL, extracted from the SHM-FMEA table on the next page**

| **Human Factors for the Team Leader** |
|---|
| Hardworking |
| Knowledge |
| Communication |
| Attention |
| Conscientiousness |
| Leader skills |
| Mental overload |
| Stress |
| Competence |
| Experience |
| Technical understanding |
| Planning skills |
| Monitoring |
| Appreciation |
| Cooperation |
| Fear |
| Management |

We can show all needed personal characteristics for the Team Leader in the following manner:

$$HF^{FMEA}_{TeamLeader} = \{Hardworking,\ Knowledge,\ Communication,\ Attention,\ Conscientiousness,\ Leader\ skills,\ Mental\ overload,\ Competence,\ Experience,\ Technical\ understanding,\ Planning\ skills,\ Monitoring,\ Appreciation,\ Cooperation,\ Fear,\ Management\} \quad (4.4)$$

**Table 9 SHF-FMEA over the Team Leader Role**

| Process Function | Potential Failure Mode | Potential Effect(s) of Failure | Potential Cause(s) / Mechanism(s) of failure | Human Factors | Detection Method(s) | Recommended Action(s) |
|---|---|---|---|---|---|---|
| Mediates between the Solution Architect and the developers | The work of the developers does not entirely match what the SA has chosen as an architecture | The architecture created from the developers cannot be matched to the one from the SA due to differences | TL is not familiar with the architecture of the SA; He has not observed the work of the developers and they have slipped from the requirements and design, selected by the SA; TL does not approve the architecture selected by SA and has a better solution; TL has decided to change a small design pattern in one place, but has no global view and that causes inconsistence | hardworking, knowledge, communication, attention, conscientiousness | When the implemented parts have to be connected; during SA, TL observation over the process; | TL being familiar with the SA's selected architecture; TL not taking alone decisions; better communication btw developers, TL and SA; |
| Lead and mentor the developers when they have problems, which cannot be mitigated by themselves alone | TL is not providing the needed help to the developers | Developers lose time and effort in solving issues, which are TL obligations | TL has too much work and obligation to fulfil; developers are not informing TL on time; TL is stubborn and tries to make the work in his own way; TL is not a good leader | communication, knowledge, leader skills | Developers take own decisions without communicating with the TL; They turn to SA for guidance and he is not adequately prepared for that; Implementations which experience lack in performance and design | TL asking constantly for questions or foggy issues; TL being more open for developers requests; TL having more time to observe personally the developers' work |
| Discussing all the details in the architecture that the SA didn't explain and in this way supporting the program specification | TL had not understood completely the proposed software architecture | TL cannot support a correct specification for the developers | TL does not have the qualities to understand the software architecture and specification | knowledge, skills | The lack of accurate specification; The lack of competent leadership from the TL | TL participating more when the architecture is being laid down by the SA; asking questions and paying attentions |
| Refines the SA's vision and makes the practical concepts clear | TL has not understood the SA vision in depth | TL is not able to refine the concepts | TL had no time to perform an in depth analysis of the SA's design and architecture; SA design is too complex | mental overload capacity, stress | The analysis is poor and the TL design innovations don't bring better performance | TL having more practice in designing solutions and applying patterns; TL communicating more with the SA |

| | | | | | | |
|---|---|---|---|---|---|---|
| The TL chooses the methodologies and techniques that will be used in a particular project | TL does not possess good technical knowledge | Developers have to manage problems on their own or turn to the SA | TL does not have enough practice; TL cannot solve problems due to lack of time; TL cannot choose proper methodology due to misleading factors or lack experience | overload, competence, experience | Developers are asking questions, which cannot be answered | TL spending more time in problem solving and communication with colleagues |
| Continuous evaluation of the solution decisions | TL is not keeping a constant track over the project | The constant evaluation is not performed | TL does not perform evaluations over the developers work due to lack of time; TL is not able to see pattern problems due to lack of knowledge; TL is not well aware of the SA architecture | conscientiousness, competence, knowledge | The proposed patterns are not correct and do not fulfill the specific project needs | TL keeps constant track of the project, observing his developers; asking questions; being curious |
| Mastery of developer skills but with conceptual vision | TL does not possess the conceptual vision to transform concepts into solutions | The concepts are not fully transformed, or are transformed improperly | TL is thinking still like a developer; his view is not wide enough; he has not mastered all skills needed to be good TL | technical understanding, competence | Visible in the design decisions he is making as well as the patterns he is choosing for the solution | TL enriching his knowledge; TL trusting on guidance by SA and colleagues |
| Direct, motivate and plan the team's work; Create an open, creative and friendly work environment | TL does not have qualities to motivate his team | Team members are working in a stressful atmosphere and are unsatisfied | TL is not skilled at leading the team; TL has not enough time to monitor team's work; TL does not appreciate team effort and creative thinking | planning skills, attention, monitoring, appreciation | Easily seen that team is not felling good and members are not well motivated | TL attending team management courses; TL paying more attention to his developers; TL having more practice in project planning |
| Take responsibility for the progress of the team's work | TL is not aware of the problems in the team | TL is not taking responsibility for progress and team | TL is not constantly speaking with the team members; TL is not keeping track over the project progress; TL is not making a proper use of all team resources | communication, cooperation, competence | Seen at meetings; Easy to notice when big problems arise | TL having time for his team members; TL making a proper planning of the resources; TL able to motivate his team |
| Manage, train and help to the development team; Conflict solving | TL is not helping the team members | Team members are having problems and this is observed in their work | TL does not have leader qualities; does not provide proper help and training due to lack of time or ideas and knowledge how to perform that; TL is not aware of all the problems | management, leader skills | Can be seen that the employees are not satisfied with their work | TL paying more attention to developers; TL having better management qualities; TL being there to protect and mentor his developers |
| Provide status reports of the team activities against the program plan; Keep the project manager informed of task accomplishment | TL is not providing periodically reports | PM is not informed for the project progress; project issues and success | TL has too many obligations; TL is afraid of saying bad news; TL has omitted his duties of reporting; TL is not reporting status due to lack of progress | fear, overload, competence | PM is not satisfied with TL's work; TL is not present at meetings or has no report | TL paying more attention to all his duties; TL not being late at telling bad news; TL always talking to PM; TL keeping track of project progress; plan and schedule |

4.2.1.3 Software Human Factors FMEA over the Business Analyst role

Analogically to the previous role here is the analysis of the Business Analyst role and of course the table with the human factors.

**Table 10 Human Factors for the BA, extracted from the SHF-FMEA table on the next page**

| **Human Factors for the Business Analyst** |
|---|
| Intelligence |
| Knowledge |
| Work overload |
| Concentration |
| Analysis skills |
| Competence |
| Communication |
| Planning |
| Openness |

We can summarize the needed personal characteristics for the Team Leader in the following manner:

$$HF^{FMEA}_{BusinessAnalyst} = \{Intelligence,\ Knowledge,\ Work\ overload,\ Concentration,\ Analysis\ skills,\ Competence,\ Communication,\ Planning,\ Openness\} \quad (4.5)$$

**Table 11 SHF-FMEA over the Business Analyst Role**

| Process Function | Potential Failure Mode | Potential Effect(s) of Failure | Potential Cause(s) / Mechanism(s) of failure | Human Factors | Detection Method(s) | Recommended Action(s) |
|---|---|---|---|---|---|---|
| Provide technical expertise(Typically in information technology applications) | BA is not providing technical expertise | Absence of expert judgment over client requirements | BA is not skilled enough to perform the needed expertise; BA has no knowledge in the researched area; BA has too much other obligations | intelligence, knowledge, work overload | Poor or no technical report; not helping the TL and PM, searching for skilled and experienced colleagues | BA being supported by other skilled colleges in case of need; BA increasing his knowledge when coming across new topic of development; attending refreshing courses |
| Understand user and other stakeholder needs and conduct requirements analysis | Not correctly understood the clients needs; unsuccessful requirements analysis | An analysis that does not satisfy clients requests; Not complete or partly useful analysis | The client is not explaining his desires directly; BA is distracted and not following the stakeholders idea; not all requirements are gathered and the analysis is not complete; BA has not the skills to perform good analysis | concentration, knowledge, analysis skills | Analysis being examined by stakeholder, other BA, manager | BA attending courses for additional technical knowledge; BA paying more attention to stakeholders requirements; having more time for a proper analysis to be created; client/BA being well prepared for the meeting |
| Identify application' solution alternatives | Associate wrong alternative; not correct identification of a solution as an alternative; | Colleagues being mislead when reading/examining the proposed alternative solutions. | Not enough knowledge to recognize the correct alternative; not familiar with the project details | knowledge, intelligence | During further work from the Architect, colleagues, manager; | BA becoming more experienced in the researched area; BA becoming familiar with the requirements; better understanding of the proposed strategies and analysis |
| Analyze existing logic with the idea to redesign and/or automate | Wrong identification of an existing system | Confusion in future work; mislead of colleagues | Not familiar with project details | knowledge, competence | Analysis being examined other employees | BA being familiar with the requirements and good understanding of the existing system |
| Recommend implementation strategies | Not correct strategies being recommended | Wasting time for re-factoring; mislead in the following choice of frameworks and architectures | Not familiar with the strategy as well as with the project | knowledge, competence | Problems in the future work, when the incompatibilities come on the surface | Better knowledge of the strategies, the requirements and the impact of the proposed solution |

| | | | | | | |
|---|---|---|---|---|---|---|
| Document recommendations to enable estimation of project scope, quality, time, cost and risks | Not all requirements are documented | Requirements analysis is not complete, further calculations of budget and time are not correctly performed | BA is not familiar with all requirements | knowledge, competence | Noticed during requirements discussion by colleagues; or manager inspection | BA being aware of requirements and details; having experience in budget and time scheduling |
| Conduct root-cause analysis of the problems | Not understanding of the potential problems | Project cost and budget are badly calculated, time is not correctly scheduled; problems are overseen or ignored | BA has no full and entire overview of the project and its properties | knowledge, competence, communication | Noticed during later planning, management, checks, budget and schedule examinations | Practice at project analysis and scheduling; attending courses; presentations |
| Develop, maintain and monitor related policies, procedures, instructions | Policies and procedures are not developed | Omitted procedures/ policies | BA has no time to perform all his duties; BA has no experience in producing policies or procedures; BA is not well familiar with the project and cannot propose new initiatives | planning, knowledge, competence | Discovered during managers check | BA having more time for his obligations; BA gathering knowledge about new practices; BA being familiar with all parts of the project |
| Reports about research findings or new business solutions | Missing such reports, which means no innovativeness | Missing of new ideas, new trends and solutions | BA has no time to do this research; BA is not innovative enough and not open for new ideas | knowledge, openness | Noticed when being inspected from the PM, or in discussions | BA having more time; attending conferences and workshops; BA observing other perspectives |

4.2.1.4 Software Human Factors FMEA over the Software Architect role

Analogically to the previous roles here is the analysis of the Software Architect role and the table with the human factors.

**Table 12 Human Factors for the SA, extracted from the SHF-FMEA table on the next page**

| **Human Factors for the Software Architect** |
|---|
| Knowledge |
| Hardworking |
| Intelligence |
| Communication |
| Competence |
| Creativity |
| Cooperation |
| Emotional stability |
| Mental overload |
| Attention |
| Judgment |
| Experience |
| Problem solving |
| Leader thinking |
| Perception |
| Professionalism |

We can summarize the needed personal characteristics for the Software Architect in the following manner:

$$HF^{FMEA}_{SoftwareArchitect} = \{Knowledge,\ Hardworking,\ Intelligence,\ Communication,\ Competence,\ Creativity,\ Cooperation,\ Emotional\ stability,\ Mental\ overload,\ Attention,\ Judgment,\ Experience,\ Problem\ solving,\ Leader\ thinking,\ Perception,\ Professionalism\} \quad (4.6)$$

**Table 13 SHF-FMEA over the Software Architect Role**

| Process Function | Potential Failure Mode | Potential Effect(s) of Failure | Potential Cause(s) / Mechanism(s) of failure | Human Factors | Detection Method(s) | Recommended Action(s) |
|---|---|---|---|---|---|---|
| Defining the software architecture | Problems/Failures in the architecture | The project is not developed as planed | SA is not completely aware of all project requirements; SA is not well familiar with the architecture; SA is not an expert in the field (does not have enough experience, knowledge) | knowledge, hardworking, intelligence, communication, competence, creativity, cooperation; emotionally stable | Detached during implementation; observation by TL or colleague SA; noticed when the selected architecture is not correctly fitted during implementation | SA being familiar with department policies, guidelines, instructions related to software development; being familiar with the organization's software architectural style |
| Derive the requirements for the software architecture | Wrong requirements or not full and comprehensive list of them | Improper architecture is designed | SA is not familiar with the requirements; SA is not aware of the architecture details | knowledge, competence, communication | Detected during examination by the TL, colleagues SA; during discussions | Training in principles and techniques for software development; Ensure all the project's technological requirements are correctly gathered, understood and properly translated for production |
| Match the software and derived requirements to the chosen architecture components and interfaces | Not correct matching; Impossibility to match the components | Requirements are not satisfied | SA is not familiar with all requirements; SA chooses an architecture that cannot correspond to the requirements | knowledge, competence, communication | Discovered during implementation; during further design and scheduling by SA or TL | SA having more time to perform the selection; SA being helped by the BA; SA spending more time with the documentation |
| Identify the key design issues for a successful development | Improper issue is selected(identified) as a key design issue | The most important issue is not resolved | SA cannot spot correctly the main issue | knowledge, competence, overload | SA is not entirely familiar with the project; SA has too many projects to manage | SA spending more time for the particular project |
| Generate alternatives and constraints for the architecture | Alternative, constraints are not generated | Wrong architecture, or no possibility for variability | SA cannot find alternatives because he is not experienced with the software technologies, standards and regulations | knowledge, competence, attention, judgment, experience | When alternative is nodded; In case the selected architecture turns out to be not effective | SA having experience with more architectures so he can propose a solutions |

| | | | | | | |
|---|---|---|---|---|---|---|
| Identify the requirements that are connected with the effectiveness and cost | Effectiveness and cost are not correctly calculated | Wrong selection of the architecture; The project runs out of budget; Not good performance | SA is not experienced in cost and time calculations; SA cannot manage and coordinate the technological services and staff | problem solving, leader thinking, intelligence, knowledge | Noticed when project is being examined by manager or the selected architecture is being checked by other SA or TL | SA having more practice in budget planning, as well as in other parts of the software development such as testing and training |
| Document, approve, and track all technological changes | Documenting is omitted | Not all changes are recorded | SA has no time to track every single detail; Not all changes have been reported; Changes happen without SA approval | overload, communication, perception, communication | Noticed later in the development process; When certain changes, are missing from the documentation | SA or colleagues keeping track of the changes; No changing without SA approval and documentation |
| Preparing risk mitigation strategies | SA has left the risk strategies to his colleagues | Risk evaluation and mitigation is not performed | SA has not enough time; SA decides to delegate issue to TL, who is not properly informed/prepared for that | overload, problem solving, communication | Records about the risk evaluation and mitigation | SA should be performing his obligations himself; in a case of delegation should be made a special plan how to act |
| Be familiar with the organization's software architectural style | SA is not familiar with the organization style | SA is implying decisions that are not following the architectural style | SA has not taken enough time to make himself familiar with the organizations' style and rules; SA is neglecting rules and proposing new ones | knowledge, hardworking, professionalism | SA's work style can be observed by the TL and Manager | SA should be working as a part of the team and the organizations and not taking alone decisions |

4.2.1.5 Software Human Factors FMEA over the Software Developer role

Here the analysis of the Software Developer and the table with the human factors.

**Table 14 Human Factors for the SD, extracted from the FMEA on the next page**

| Human Factors for the Software Developer |
|---|
| Hardworking |
| Knowledge |
| Persistence |
| Concentration |
| Intelligence |
| Attention |
| Competence |
| Personal overload |
| Dutifulness |
| Communication |
| Cooperation |
| Motivation |
| Achievement |
| Responsibility |
| Talkativeness |
| Coordination |
| Personal organization |

The summarized personal characteristics for the Software Developer look like the following:

$$HF^{FMEA}_{SoftwareDeveloper} = \{\textit{Hardworking, Knowledge, Persistence, Concentration, Intelligence, Attention, Competence, Personal overload, Dutifulness, Communication, Cooperation, Motivation, Achievement, Responsibility, Talkativeness, Coordination, Personal organization}\} \quad (4.7)$$

**Table 15 SHF-FMEA over the Software Developer Role**

| Process Function | Potential Failure Mode | Potential Effect(s) of Failure | Potential Cause(s) / Mechanism(s) of failure | Human Factors | Detection Method(s) | Recommended Action(s) |
|---|---|---|---|---|---|---|
| Designs different software components | The designed elements are not correct or do not follow the requirements | The produced code is not fully operational | SD is not familiar with all requirements; SD has not tested; SD is not asking questions in case of problems; SD is not skilled in programming language and logic | hardworking, knowledge, persistence | Can be seen in the code – bugs or other problems; lack of fulfilled requirements will show at later phase - testing; quality control | SD having good programming skills; SD sharing problems with colleagues; SD being familiar with the requirements and paying attention to debugging |
| Approve software only if sure that it is safe, meets the specifications, has passed appropriate tests, and is not a threat for the life or the environment | Neglect obligations like inspecting the code or checking if all requirements are met | Software is stated as approved and according to the specification though it isn't | SD is distracted by something; SD omits debugging; SD is having too much work or too little time; SD is not being concentrated; SD is not good skilled in programming and testing | concentration, knowledge, intelligence | Bugs and problems can be seen in the code; problems are spotted by testers, QA | SD paying more attentions to his work and requirements; SD having more time to look things up |
| Strive to fully understand the specifications for software on which they work | Not attempting to understand all requirements | Requirements are not all understood and specification is not familiar to SD | SD is not paying attentions to requirements and specification with the idea that TL will tell them what to do; SD is omitting readying and understanding the specification; SD has no time to read the specification | attention, concentration, competence, overload | Obvious in meetings with TL; obvious in case SD has to think of a decision to a question | SD paying more attention to his obligations; SD having time to perform an in-depth analysis |
| Ensure adequate testing, debugging, and review of software and related documents | Proper testing and debugging is missing; paper work is skipped | Documents are not created and the proper testing and review of software is not performed | SD is having to many obligations and have no time to perform this one; SD is bored to perform paper work | overload | Obvious that the documents are not written; code is not good tested | SD having more time for all tasks; SD being motivated to make his paper work and review of code |
| Maintain the integrity of data, being sensitive to outdated or flawed occurrences | Not paying great attention to data management | Integrity of data may be lost | SD is not skilled at data management; SD omits duties to manage data; SD is careless about outdated or flawed occurrences | skills, knowledge | Lost integrity of data is hard to spot but when found difficult to fix | SD being careful and experienced at data management |
| Take responsibility for detecting, correcting, and reporting errors in software and associated documents | Neglecting obligations as bug detecting and tracking | Errors and bugs are not corrected and not documented | SD is bored and not motivated to search for errors; SD has no time to document each error; SD is pointing finger at the tester for looking and documenting the errors | concentration, overload, dutifulness | Not taking responsibility in front of TL; lack of errors report | SD having more time for his responsibilities; SD being motivated in bug searching, fixing and documenting |

| | | | | | | |
|---|---|---|---|---|---|---|
| Integrating software modules into software components and units | Not correctly integrating all parts | Components are put together correctly but are not fully operational | SD is not skilled at the specific programming language; SD has not made sure his code will work with those of his colleagues | competence, knowledge, communication | Visible when trying to put all parts together | SD talking more to colleagues; SD paying attentions to others' code |
| Assigned full or part time to participate in project team activities | Not participating in team activities | SD is left outside of the team and is not sharing the team spirit | SD is not social; SD has too much work; SD is not interested in communicating with others | communication, cooperation, | SD is not attending team meetings; team-building | SD trying to be more social; SD attending team activities |
| Responsible for contributing to overall project objectives and specific team deliverables | Not contributing to project activities | Specific deliverables are not performed | SD is careless in his work; does not perform his duties; SD is not motivated | motivation, concentration | Easy to spot that SD's is not effective and motivated in his work | SD being more careful in his work; TL can find different ways to stimulate the SD to give his best |
| Participates with TL in application documentation | Does not cooperate with the TL | TL is left alone to do all the documentation | SD has no time for this responsibility; SD is not willing to do paper work; SD has no good style at making such documents | overload, achievement, competence | TL is making all the work himself; TL is not receiving help from SD | SD cooperating with the TL; SD being motivated to work together with the TL |
| Designs, codes, and builds the application | Designing and coding are not performed on high level | The code is full of errors; bad performance; not following the requirements | SD is not a skilled developer; SD is not familiar with the requirements; SD does not consult with his colleagues or TL and works alone | knowledge, competence, intelligence, communication | Bad code can be easily discovered by inspection; Not meeting the requirements is also obvious in later checks | SD being more experienced; SD working better with colleagues; SD paying more attention to his work |
| Participates in code reviews and testing | Not performing his duties by testing and reviewing | QA is left out to test by himself and with no help from SD | SD is not having time for helping colleagues; SD is not willing to help; SD has too much other obligations | cooperation, communication | Lack of desire to help is easy to spot and difficult to tolerate | SD working better with colleagues and being eager to help |
| Fixes bugs, defects, and shortcomings | Omits testing and bug fixing | Code is left without fixing | SD is not having time; throwing responsibility to QA; SD is not good at bugs detecting and fixing | cooperation, responsibility, knowledge, competence | Bugs in the code are found during QA testing | SD being more precise in his work; SD paying more attentions to bug fixing |
| Work with colleagues within the designated project guidelines | Not being friendly and cooperative | SD is not easy to work with and is not a good team player | SD is not friendly; prefer working alone; does not socialize with colleagues | social contact, communication, talkativeness | It is obvious in his lack of communication and cooperation | SD trying to socialize; perform better in team work in order to fulfill project needs |
| Notify the TL of any expected difficulties or issues arising | Trying to resolve problems on his own | TL is left not notified of the problems that have occurred; SD is making decisions that may not be of his competence | SD is believing it is in his authority to answer such questions; SD does not want to bother TL; SD is feeling proud and independent to manage with issues on his own, neglecting teams procedure of informing | competence, coordination, organization | Difficult to spot, may be seen later, when the problem becomes really big and eventually SD has to inform his TL | SD should know his place in the team and always inform TL in case of a major issue |

4.2.1.6 Software Human Factors FMEA over the Software Tester role

Here the analysis of the Software Tester and the table with the human factors.

**Table 16 Human Factors for the ST, extracted from the SHF-FMEA on the next page**

| **Human Factors for the Software Tester** |
|---|
| Competence |
| Knowledge |
| Communication |
| Personal attitude |
| Motivation |
| Overload |
| Concentration |
| Understanding |
| Coordination |
| Too high self-confidence |
| Creativity |
| Imagination |
| Open minded |
| Self-organization |

The summarized personal characteristics for the Software Tester look like the following:

$$HF^{FMEA}_{SoftwareTester} = \{Competence,\ Knowledge,\ Communication,\ Personal\ attitude,\ Motivation,\ Overload,\ Concentration,\ Understanding,\ Coordination,\ Self\text{-}confidence,\ Creativity,\ Imagination,\ Open\ minded,\ Self\text{-}organization\} \quad (4.8)$$

**Table 17 SHF-FMEA over the Software Tester Role**

| Process Function | Potential Failure Mode | Potential Effect(s) of Failure | Potential Cause(s) / Mechanism(s) of failure | Human Factors | Detection Method(s) | Recommended Action(s) |
|---|---|---|---|---|---|---|
| Work with the QA to build a test strategy and test plans | Missing tests and wrong test strategy | Undiscovered problems, which on later step will cost very expensive to be mitigated | Lack of communication btw the Tester and the QA; Not able to agree on the needed strategy; lack of knowledge about the needed testing | competence, knowledge, communication, personal attitude | In meetings where the testing strategy is discussed | More communication btw the team members that have this obligation; Teaching seminars in order to get new knowledge in the area |
| Designing and implementing the test scripts and test cases | False test scripts and test cases | Inefficient testing which ends with undiscovered problems | Lack of knowledge which leads to incomplete and inefficient testing; Lack of time for full testing; Lack of motivation | competence, knowledge, personal attitude, motivation | Discussions about the test scripts and cases; Inspections from the TL; Later when the application is not working as expected | Discussions with the TL; Enough time for testing; Seminars and motivation from the TL |
| Functional analysis of the software application in the actual environment | Some steps in this functional analysis are not identified correctly or omitted during testing | There are steps in the functional analysis left untested and this may lead to some wrong functionality or errors | The Tester has not time to test everything; He has not prepared a functional testing strategy | overload, competence, concentration | Detected when the software is not working as expected in the real environment | Being careful and performing test on each functional part in the concrete environment; The ST expanding his view to predict what may go wrong and perform the necessary testing steps |
| Design, specification and implementation of the test environment and the test-data | Wrong specified test environment and test-data | The software cannot be correctly tested; not all problems are discovered and mitigated | Not enough knowledge and experience with the needed techniques; Wrong identified test-data; Little time | overload, competence, knowledge | Detected later on when problems occur; it is possible that some errors stay undiscovered | Paying more attention and more time for designing the environment and the data-sets; Additional learning |
| Understanding of the software development process, of the operating system and the network infrastructure that are used for deployment of the software | The T does not understand in depth the development process or the complete architecture of the used network | In the ready for installation program may not be considered some limitations implied by the network or by some specifics of the software design | The ST does not have a good understanding of software development; May be : not attending team meetings; be unfamiliar with network architecture; be unfamiliar with the architecture on which the software is build | knowledge, understanding, coordination | Observed when the software is not proceeding properly and cannot be deployed | Paying great attention to development details as well as to the network infrastructure |

| | | | | | | |
|---|---|---|---|---|---|---|
| Execute the tests, document the results and maintain the records | Wrong tests, wrong results and lack of documentation of the whole process | Undiscovered failures in the software; lack of documentation that could be used in the next testing process | Not enough knowledge how to build the tests; Lack of time and desire to write a documentation | knowledge, overload, motivation | Detected on meetings when discussing the testing progress and the documentation; Detected later on when evaluating the results. | The ST have to pay more attention in the testing and documenting process; Has to put more effort in achieving the software goals |
| Be familiar with similar type of software, its complexity and typical functionality | ST is not well familiar with other software products of the type | ST cannot use experience from similar projects and it is possible that he oversees some problems | ST has no time to search for other similar solutions with ready testing process and prefers to build it on his own, but conducts failures | overload, too high self-confidence | Difficult to discover, but the problems come on later step when evaluating the testing process | Analysis of common systems in the field that can be used as basis for the current testing process |
| Being familiar with the latest standards, tools and methods that can be used in the testing process | ST is not constantly enriching his knowledge in the area | New standards or methods may be new for him | ST is not learning new techniques due to being old fashioned or unmotivated; T is missing new items and tools that will make his work easier | creative, imagination, open minded | Can be discovered only other colleagues criticize his work | ST should be constantly looking for new information and new ideas in order to use the most trendy solutions |
| Perform defect tracking, status reporting and auditing | ST is not continuously tracking the current software system | Threats, defects may remain undetected or untraced in the documentation reporting | ST is not having time to perform new defects search; T is postponing tasks for tracking and reporting due to not being motivated | overload, organization, motivation | Lack of written reports, defect tracking and auditing is obvious | More control over the ST's obligations so that he performs defect tracking and status reporting |
| Retesting after fixing problems | ST is not performing retesting, or only a part | New failures | ST is not having time to perform the retesting or he does not have the desire to do that | overload, motivation | Observed with the occurrence of new problems in the software | Control and motivation over the testing team |

4.2.1.7 Software Human Factors FMEA over the Software Quality Engineer role

Here the analysis of the Software Quality Engineer and the table with the human factors.

**Table 18 Human Factors for the SQE, extracted from the SHF-FMEA on the next page**

| **Human Factors for the Software Quality Engineer** |
|---|
| Overload |
| Coordination |
| Communication |
| Competence |
| Knowledge |
| Over self-confidence |
| Planning |
| Attention |
| Intelligence |
| Understanding |
| Patience |
| Friendliness |
| Concentration |
| Professionalism |
| Cooperation |

The summarized personal characteristics for the Software Quality Engineer look like the following:

$$HF^{FMEA}_{SoftwareQualityEngineer} = \{Overload,\ Coordination,\ Communication,\ Competence,\ Knowledge,\ Self\text{-}confidence,\ Planning,\ Attention,\ Intelligence,\ Understanding,\ Patience,\ Friendliness,\ Concentration,\ Professionalism,\ Cooperation\} \quad (4.9)$$

**Table 19 SHF-FMEA over the Software Quality Engineer Role**

| Process Function | Potential Failure Mode | Potential Effect(s) of Failure | Potential Cause(s) / Mechanism(s) of failure | Human Factors | Detection Method(s) | Recommended Action(s) |
|---|---|---|---|---|---|---|
| Planning and implementing a product testing regime during the development and construction process | Planning and implementation of the testing scenarios are not performed | The test regime is being developed after the code has been written and this affects the whole development process | QE is having too much obligations; QE is not being helped by ST and Architect for the project requirements; QE does not have enough experience | overload, coordination, communication, competence, knowledge | Lack of performed test cases is obvious in the number of bugs | QE having more time for his duties; QE being helped and monitored by SA or TL; QE being skilled in planning and implementing of testing regimes |
| Responsible for guaranteeing a quality level for the end client | Not taking responsibility and pointing finger at the development team | Bad atmosphere in the team due to QE's desire not to take responsibility | QE not admitting his mistakes; QE not familiar with all his obligations; QE not able to plan all needed actions | self-confidence, competence, planning | Can be seen at team meetings | QE being able to admit being wrong and taking responsibility for his work |
| Understand the requirements of the project's technological scope, its required functionality and quality grade | Not all requirements (functionality and quality) are met | Not all requirements are checked, tested and inspected, resulting in product being not fully operational like specified | QE is not familiar with requirements due to not attending team meetings; not reading specification; not checking what is written and interpreting it on his own; not getting proper explanations | coordination, attention, intelligence, knowledge, understanding | Can be seen at a later phase by testers; or even users | QE making effort to be familiar with requirements; paying attention when being explained about details, value of project |
| Assuring the needed level of quality in the completed objectives | Time is pressing the team so some tests are omitted | Parts of the development are left not inspected in depth, hidden bugs may have remained | QE is really pressed by the time; QE is not being patient to perform each test again and again; The code is not well introduced by SDs | overloaded, patience, coordination | Lack of proper quality level is visible in testing as well as on a later phase by the user/client | Better schedule of all properties; all team members working according it; Better control by the TL |
| Works with the Business Analyst and the Software Architect to convert the requirements and design documents into a set of testing cases and scripts | Not good communication with BA and SA; Requirements are not good transformed into test cases and scripts | The produced tests are not useful and do not meet project level and details; The project is not correctly tested and not all client needs are satisfied | QE is not social and is not communicating with colleagues; QE is pretending to know all and makes tests and analyses on his own; QE not being familiar with the requirements | friendliness, coordination, communication, competence | Can be seen when the project is not meeting the client requirements; The communication level in the team is not good and the atmosphere is not productive | QE being more social and providing better work atmosphere in team; QE carefully reading and examining requirements when transforming them to test cases |

| | | | | | | |
|---|---|---|---|---|---|---|
| Performs random testing of all components to check again for errors in the system | Not performing random testing and relying only on the testers | A random bug may not be found | QE does not have time for this testing; QE does not know how to perform this random testing | overload, knowledge, competence | Can be discovered or not, depending on how random the bug is | QE making all diversity of tests so he can spot the bug or at least the situation in which it may show up |
| Measurement and quantification of the completed solution performance | Not making performance tests | The performance of the solution is not measured and could be quite low | QE has no time to measure quality due to bad schedule; QE may not be familiar with the tests for performance | overload, knowledge, competence | Bad performance can be seen later, when the solution is brought to the clients | QE having time and skills for performance testing |
| Be familiar with the organization's software architectural style, departmental testing policies, criteria, strategy and procedures | QE is not familiar with organization testing policies and software strategies | QE's way of testing and documenting does not meet the organization’s expectation | QE is not introduced to the specific working style; QE has problems to work according to the organization’s politics | coordination, knowledge, competence | Can be seen if he is not keeping the organization’s rules or practices and is making decisions on his own | QE should be given time to become familiar with the organization’s testing practices and development style |
| Being familiar with the latest standards | Not familiar with the needed standards and testing technologies | The work of QE is not compliant with standards and technologies; does not follow organization’s politics | QE is not paying attention to the latest standards and technologies due to the fact that he is not familiar with them or he does not agree with them and has other point of view | concentration, professionalism | Obvious in his work; can be observed his way of making things and taking decisions | QE should follow project and organization’s politics for making decisions, should be familiar with new designs and standards, associated to the project |
| Provide advice and guidance on quality-issues when and where needed | QE is not providing proper help when asked | Those looking for QE’s help are left with questions | QE does not have time for such questions; QE is not a helpful person and is avoiding communicating with colleagues | overload, character, communication, cooperation | Can be observed in the everyday communication and cooperation process | QE should be ready to discuss and help his colleagues; Team building |

## 4.3 Summary over the Software Human Factors FMEA

We have conducted the innovative adoption of the FMEA as Software Human Factor FMEA in a strictly analytical way over the responsibilities of the software development team members, explained in Chapter 3, and were able to find out all different human features that stay behind the different failures or potential problems in the software development process. Here we will show once again all the factors for the different roles and after this we will put them together in order to gain the full list of human factors critical for the software engineering process.

$$HF^{FMEA}_{ProjectManager} = \{\textit{Coordination, Self-management, Stress, Competence, Knowledge, Effectiveness, Concentration, Communication, Self-development, Liberalism, Control, Egoism, Confidence, Organization}\} \quad (4.10)$$

$$HF^{FMEA}_{TeamLeader} = \{\textit{Hardworking, Knowledge, Communication, Attention, Conscientiousness, Leader skills, Mental overload, Competence, Experience, Technical understanding, Planning skills, Monitoring, Appreciation, Cooperation, Fear, Management}\}$$

$$HF^{FMEA}_{BusinessAnalyst} = \{\textit{Intelligence, Knowledge, Work overload, Concentration, Analysis skills, Competence, Communication, Planning, Openness}\}$$

$$HF^{FMEA}_{SoftwareArchitect} = \{\textit{Knowledge, Hardworking, Intelligence, Communication, Competence, Creativity, Cooperation, Emotional stability, Mental overload, Attention, Judgment, Experience, Problem solving, Leader thinking, Perception, Professionalism}\}$$

$$HF^{FMEA}_{SoftwareDeveloper} = \{\textit{Hardworking, Knowledge, Persistence, Concentration, Intelligence, Attention, Competence, Personal overload, Dutifulness, Communication, Cooperation, Motivation, Achievement, Responsibility, Talkativeness, Coordination, Personal organization}\}$$

$$HF^{FMEA}_{SoftwareTester} = \{\textit{Competence, Knowledge, Communication, Personal attitude, Motivation, Overload, Concentration, Understanding, Coordination, Self-confidence, Creativity, Imagination, Open minded, Self-organization}\}$$

$$HF^{FMEA}_{SoftwareQualityEngineer} = \{\textit{Overload, Coordination, Communication, Competence, Knowledge, Self-confidence, Planning, Attention, Intelligence, Understanding, Patience, Friendliness, Concentration, Professionalism, Cooperation}\}$$

Summarizing these factors into one with the help of the following formula:

$$HF^{FMEA}_{SoftwareProcess} = \{HF^{FMEA}_{ProjectManager}, HF^{FMEA}_{TeamLeader}, HF^{FMEA}_{BusinessAnalyst}, HF^{FMEA}_{SoftwareArchitect}, HF^{FMEA}_{SoftwareDeveloper}, HF^{FMEA}_{SoftwareTester}, HF^{FMEA}_{SoftwareQualityEngineer}\} \quad (4.11)$$

and after merging them and taking out the repeated ones we have ended with the following list of human factors or characteristics that influence the software development performance.

1. Coordination
2. Self-management
3. Mental Overload=Stress
4. Competence
5. Knowledge
6. Effectiveness
7. Concentration
8. Communication
9. Self-Development
10. Liberalism
11. Control delegation
12. Selfish=Egoism
13. Over self-confident
14. Self-organization
15. Hardworking
16. Attention
17. Conscientiousness
18. Leader skills
19. Experience
20. Personal grow
21. Understanding ability
22. Planning skills
23. Observing ability
24. Appreciation
25. Cooperation
26. Fear
27. Management skills
28. Intelligence
29. Analysis skills
30. Openness
31. Creativity
32. Emotional stability
33. Judgment
34. Problem solving ability
35. Perception
36. Professionalism
37. Persistence
38. Dutifulness
39. Motivation
40. Achievement
41. Responsibility
42. Talkativeness
43. Personal attitude
44. Technical understanding
45. Imagination
46. Patience
47. Friendliness

Having all the critical human factors for the software process we were faced with a new problem. How can we measure these traits and how can we examine a person in order to be able to understand which features does he posses and into which extent so that we can find out how they influence his work performance.

We will manage with this challenge in the next two chapters. First we will adopt a well-known psychological method in order to measure the personal features and then a special statistical method in order to find out how they influence the individuals' performance.

# 5 Chapter – Definition and Evaluation of the IT Human Factors

In the previous chapter was introduced the big number of human factors for the software development team members gathered by the adopted FMEA analytical approach. Here we had the challenge to find out how we can measure these human factors in a way that we can find the connection between the factors and the individual performance. After long research we have decided to adopt the ‘’Big Five’’ theory, very widely used in the recruitment and personnel selection process, in order to be able to evaluate all these factors and to find the connection with the individual’ performance. Adopting this method for our need we were able to measure the specific personal traits and the personal productivity and we are using this information in the next chapter in order to discover the dependence between the human characteristics and the productivity.

## 5.1 The five personal features

The Big Five model is a comprehensive, data-driven approach that evaluates five different compound personal traits in order to build a complete psychological profile. The five factors were discovered and formulated by several independent researchers and had a long maturing process, summarized by Digman (Digman 1990).

The first idea about analyzing the human personality came in the beginning of the 20-th century from McDougall (McDougall 1932) but the first version of the model was proposed by Ernest Tupes and Raymond Cristal in 1961 (Tupes & Cristal 1961). Anyway this proposal reaches the academic audience twenty years later and in this time there were already other scientific papers proposing similar ideas. In 1990 Digman emerges the five factor model and few years later Goldberg refines it to the highest level (Goldberg 1993). The interesting point in the history of the Big Five is that the personal features were discovered from the different scientists to be the same, and although that there are some differences all come to the decision that particularly these five features with their facets (John, Robins & Pervin 2008) describe the human behavior in the best way. The Big Five traits are also referred to as the "Five Factor Model" or FFM (Costa & McCrae 1992) and as the Global Factors of personality (Russell & Karol 1994). The Big Five factors are Openness, Conscientiousness, Extroversion, Agreeableness and Neuroticism (OCEAN). Sometimes the neuroticism element is called Emotional Stability as well the openness factor is named Intellect. Here we give short explanation of these traits.

**Openness to experience / Intelligence**
*(inventive / curious vs. cautious / conservative)*

Openness, in some places named also Intelligence is the ability of the people to accept and to search for new ideas, knowledge, experience and so on. It describes the originality and complexity of an individual and distinguishes the imaginative from the down-to-earth people. (John, Robins & Pervin 2008) Such persons are ready for new experience, intellectually searching and impressed by art. People with low levels of openness are traditional and have conventional understandings.

**Conscientiousness**
*(efficient / organized vs. easy-going / careless)*

This is a feature that expresses self-discipline and determination and desire for achievement. It expresses an intention to behave in a planned matter, goal-directed and thinking before acting. Such people follow norms and rules, they are always on time, study hard and give their best in the job. They are not impulsive and show high values of thoughtfulness. (John, Robins & Pervin 2008) Low levels of conscientiousness mean unorganized people, that don't really care how they are performing in their job and don't feel responsible for their actions.

**Extroversion**
*(outgoing / energetic vs. shy / withdrawn)*

Extroversion can be described by positive emotions, desire to seek for stimulations and company of others. It is an energetic and positive attitude to the world and is described with features like: sociability, activity, assertiveness, and positive emotionality. For these people it is easy to approach strangers, to introduce themselves, to be the leader and the centre of a company. (John, Robins & Pervin 2008) When being around people they like to talk, put themselves forward and keep the attention. Introverts lack the social cheerfulness and activity levels of the extroverts. They tend to be quiet and less interested in the social world.

**Agreeableness**
*(friendly / compassionate vs. competitive / outspoken )*

This feature is expressed in compassionate and cooperative behavior. It shows a pro-social and communal orientation toward others and can be described with traits like: altruism, tender-mindedness, trust and modesty. (John, Robins & Pervin 2008) This characteristic is very important for the social harmony and understanding. Such people are generous, kind, friendly, caring, cooperative and ready to compromise their own interests. People with low level of agreeableness put first their own-interest and show features like: suspicion, unfriendliness and uncooperativeness.

**Neuroticism**
*( sensitive / nervous vs. control / confident)*

Neuroticism is characterized with the propensity to negative emotions like anger, nervousness and depression. It contrasts emotional stability and is expressed with emotions like: feeling anxious, nervous, sad, and tense. (John, Robins & Pervin 2008) People with high score of neuroticism tend to accept ordinary situations as threading and small obstacles as hopelessly difficult. They are in negative emotional states for long time and this influences their working process. Persons with low neuroticism are not so easily disturbed and emotionally stable.

We are introducing here a table with three different approaches for the Big Five and their facets, summarized by Oliver John and his colleagues in their book: (John, Robins & Pervin 2008). These facets bring additional understanding for the big five traits and will help us on the next step when matching the discovered software human factors to the Big Five.

**Table 20 Defining Facets for the Big Five Trait Domains (John, Robins & Pervin 2008)**

| Lexical facets (Saucier & Ostendorf 1999) | NEO-PI-R facets (Costa & McCrae 1992) | CPI-Big Five facets (Soto & John 2008) |
|---|---|---|
| Extraversion (E) facets<br>E Sociability<br>E Assertiveness<br>E Activity/Adventurousness<br>E Unrestrained<br>*[A Warmth/Affection]* | E Gregariousness<br>E Assertiveness<br>E Activity<br>E Excitement-Seeking<br>E Positive emotions<br>E Warmth | E Gregariousness<br>E Assertiveness/Leadership<br>*[O Adventurousness]*<br>E Social Confidence vs. Anxiety |
| Agreeableness (A) facets<br>A Warmth/Affection<br>A Modesty/Humility<br>A Generosity<br>A Gentleness | *[E Warmth]*<br>A Modesty<br>A Trust<br>A Tender-Mindedness<br>A Compliance<br>A Straightforwardness | A Modesty vs. Narcissism<br>A Trust vs. Suspicion<br>A Empathy/Sympathy<br>A Altruism |
| Conscientiousness (C) facets<br>C Orderliness<br>C Industriousness<br>C Reliability<br>C Decisiveness<br>*[O Perceptiveness]* | C Order<br>C Achievement Striving<br>C Dutifulness<br>C Self-Discipline<br>C Competence<br>C Deliberation | C Orderliness<br>C Industriousness<br>C Self-Discipline |
| Neuroticism (N) facets<br>N Insecurity<br>N Emotionality<br>N Irritability | N Anxiety<br>N Angry Hostility<br>N Depression<br>N Self-Consciousness<br>N Vulnerability<br>N Impulsiveness | N Anxiety<br>N Irritability<br>N Depression<br>N Rumination–Compulsiveness<br>*[E Social Confidence vs. Anxiety]* |
| Openness (O) facets<br>O Intellect<br>O Imagination/Creativity<br>O Perceptiveness | O Ideas<br>O Aesthetics<br>O Fantasy<br>O Actions<br>O Feelings<br>O Values | O Intellectualism<br>O Idealism<br>O Adventurousness |

‘’Some facets (e.g., CPI Adventurousness) are listed once under their primary Big Five domain (e.g., Openness) and again in brackets under another Big Five domain if their best-matching facet appears there (e.g., next to NEO Excitement-Seeking, which is an Extraversion facet on the NEO-PI-R but also has a substantial secondary correlation with Openness).’’ (John, Robins & Pervin 2008) In (John, Robins & Pervin 2008) - table 4.4 can be observed another detailed list with the Central Trait Adjectives for the Five Factors.

## 5.2. Matching between the Big Five traits and the IT Human Factors

The Big Five Trait Domain that we are adopting in our method is the NEO-PI-R, as with its 30 facets is the most comprehensive one. Based on the analysis from chapters 2, 3 and 4 we have found the personal competencies and the special human factors that influence the individuals' performance. Having these critical human factors for the software process we were faced with the problem: how can we measure them? For this purpose we have used the following matching between the critical human factors and the Big-Five psychological traits. This matching helps us to evaluate the human traits and in this way to observe the dependence between them and the performance.

In the following part we will show the matching between the human factors that we have found and the Big Five traits.

**Table 21 Matching between the Big Five traits and the Software Human Factors**

| NEO-PI-R facets (Costa & McCrae 1992) | Human Factors important for the software development process |
|---|---|
| <u>Extraversion (E) facets</u><br>E Gregariousness<br>E Assertiveness<br>E Activity<br>E Excitement-Seeking<br>E Positive emotions<br>E Warmth | Communication<br>Selfish=Egoism<br>Over self-confident<br>Leader skills<br>Management skills<br>Talkativeness<br>Judgement |
| <u>Agreeableness (A) facets</u><br>A Modesty<br>A Trust<br>A Tender-Mindedness<br>A Compliance<br>A Straightforwardness | Liberalism<br>Appreciation<br>Cooperation<br>Problem solving<br>Perception<br>Persistence (by low A)<br>Friendliness |
| <u>Conscientiousness (C) facets</u><br>C Order<br>C Achievement Striving<br>C Dutifulness<br>C Self-Discipline<br>C Competence<br>C Deliberation | Coordination<br>Self-management/ organization<br>Control delegation<br>Effectiveness<br>Hardworking<br>Attention<br>Planning skills<br>Professionalism<br>Dutifulness<br>Achievement<br>Responsibility |
| <u>Neuroticism (N) facets</u><br>N Anxiety<br>N Angry Hostility<br>N Depression<br>N Self-Consciousness<br>N Vulnerability<br>N Impulsiveness | Mental Overload; Stress<br>Concentration<br>Fear<br>Emotional stability<br>Personal attitude<br>Patience |
| <u>Openness (O) facets</u><br>O Ideas<br>O Aesthetics<br>O Fantasy<br>O Actions<br>O Feelings<br>O Values | Self-development<br>Personal growth<br>Understanding ability<br>Observing ability<br>Intelligence<br>Analysis skills<br>Creativity<br>Imagination |

After the matching process was over we have found few additional features that don't pass into the Big Five traits and we have decided to include them as additional factors. These are *the Experience and the Motivation.*

Under Experience we have the following sub-traits: - competence; - knowledge and - technical understanding.

As the values for the Big Five are in percentage, we have decided also to use percentage for the additional factors. In order to estimate the value of the Motivation we have used special questions, shown in table 24 and have evaluated them in the same manner like the Big Five test. For evaluation of the Experience we have taken a 20 years basis for 100% and we have calculated the values based on that.

The last and the most important factor that we have evaluated and that is for us the end goal was *the Performance*. In order to evaluate it we have used again several sources: first the own evaluation, then this of the supervising personnel/manager and last but not least the evaluation of the colleagues. In this manner we were able to calculate the value of the Performance/Productiveness also in percentage of the managed work per month. So we can summarize the seven factors that we decided to investigate in connection with the individual performance, and they are:

1. Openness
2. Conscientiousness
3. Extroversion
4. Agreeableness
5. Neuroticism
6. Experience
7. Motivation

## 5.3 The evaluation test

In order to measure the listed above seven personal characteristics we have adopted the Big Five questions and have added additional ones in order to evaluate the other two factors and also the approximate performance.

First we will have a look over the standard questions, shown in the following Table 22 and then we will take a look over the additional ones.

The table shows all positive and negative questions for the Big Five traits. The questions are taken from an on-line pool for scientific collaboration ''International Personality Item Pool'' (International Personality Item Pool 1997).

**Table 22 Big Five Questions (International Personality Item Pool 1997)**

| Positive Questions | Negative Questions |
|---|---|
| **Extraversion** | |
| I am the life of the party. | Don't talk a lot. |
| Feel comfortable around people. | Keep in the background. |
| Start conversations. | Have little to say. |
| Talk to a lot of different people at parties. | Don't like to draw attention to myself. |
| Don't mind being the center of attention. | I am quiet around strangers. |
| **Agreeableness** | |
| I am interested in people. | I am not really interested in others. |
| Sympathize with others' feelings. | Insult people. |
| Have a soft heart. | I am not interested in other people's problems. |
| Take time out for others. | Feel little concern for others. |
| Feel others' emotions. | |
| Make people feel at ease. | |
| **Conscientiousness** | |
| I am always prepared. | Leave my belongings around. |
| Pay attention to details. | Make a mess of things. |
| Get chores done right away. | Often forget to put things back in their proper place. |
| Like order. | Shirk my duties. |
| Follow a schedule. | |
| I am exacting in my work. | |
| **Emotional Stability** | |
| Am relaxed most of the time. | Get stressed out easily. |
| Seldom feel blue. | Worry about things. |
| | I am easily disturbed. |
| | Get upset easily. |
| | Change my mood a lot. |
| | Have frequent mood swings. |
| | Get irritated easily. |
| | Often feel blue. |
| **Openness/Intelligence** | |
| Have a rich vocabulary. | Have difficulty understanding abstract ideas. |
| Have a vivid imagination. | I am not interested in abstract ideas. |
| Have excellent ideas. | Do not have a good imagination. |
| I am quick to understand things. | |
| Use difficult words. | |
| Spend time reflecting on things. | |
| I am full of ideas. | |

We can see that we have 10 questions pro Factor and they can be categorized into positive or negative one. Every question has 5 possibilities for an answer: Very Inaccurate, Moderately Inaccurate, Neither Inaccurate nor Accurate, Moderately Accurate, Very Accurate. Depending on the question type – positive or negative - from 1 to 5 points are given. The table for the evaluation looks like this:

**Table 23 Points for the different answers**

| Answer | Points for statement | |
|---|---|---|
| | Positive | Negative |
| **Very Inaccurate** | 1 | 5 |
| **Moderately Inaccurate** | 2 | 4 |
| **Neither Inaccurate nor Accurate** | 3 | 3 |
| **Moderately Accurate** | 4 | 2 |
| **Very Accurate** | 5 | 1 |

Having the standard Big Five questions, let us take a look now over the additional ones. They are listed in the table below and are taken from a position paper about the Behavior-based Assessment (Smolders et al. 2009) and help us to evaluate the Motivation and Experience factors.

**Table 24 Additional questions for ‚Experience and Motivation' (Smolders et al. 2009)**

| Questions for the factors Experience and Motivation | |
|---|---|
| **Motivation** | **Experience** |
| You feel the goals you are supposed to achieve are realistic and attainable? | What is your current working position? |
| Feedback from your manager/supervisor is clear and directed at improving your performance? | What is your age? |
| Your job is both interesting and challenging? | How many years have you worked at your current position? |
| You feel that your current salary motivates you to perform? | |
| The advancement and growth opportunity within the organization motivates you to perform better? | |
| You receive recognition for your achievements from your manager/supervisor? | |
| You receive ongoing training to improve your ability and skills? | |
| Your manager/supervisor lets you take responsibility for the tasks you perform? | |
| Your current performance appraisal system motivates you to achieve your goals and improve your performance? | |

The additional questions about Motivation are answered like the previous ones, like shown in table 23 and the questions about Experience are being answered with plain explanation text. The questions in their actual form in the test were randomized and this is due to the fact that if answered one after the other from a particular type, they tend to seem the same and a person can simply copy the previous statement without thinking on the current one.

There is one more very important question that was included in the test and this is: *With what percentage would you estimate your everyday performance?* As already said in order to measure this, we have used the personal evaluation and this from the supervising head and from the colleagues for each examined person. In addition to the self-estimation we have asked separately the supervisors/managers and the colleagues how they will evaluate the work of the examined person in *succeeded amount of work per month*. Having all these questions we were able to build our test and to distribute it around different software companies. We have used an online platform (Zoho Challenge 2010), so that it was easy to access, fill and evaluate.

We have distributed the questionnaire in five companies and from 200 participants we have gained 73 usefully filled tests. Then we have summarized the data (as there were a lot of tests that had identical results) and we have presented it in the following Table 25.
The people that fulfilled the test were between 26 and 55 years old with different experience on the current position (20 years=100%). The number of the people according to the positions that they have looks like follows:

Project Manager - 6
Business Analyst - 10
Software Architect - 10
Team Leader - 10
Software Developer - 15
Quality Engineer - 10
Software Tester – 12

**Table 25 Summarized data from the test-results**

| Performance [%] | Motivation [%] | Conscientiousness [%] | Openness [%] | Agreeableness [%] | Experience [%] | Extraversion [%] | Emotional stability [%] |
|---|---|---|---|---|---|---|---|
| 46 | 30 | 36 | 58 | 40 | 20 | 46 | 78 |
| 47 | 34 | 36 | 60 | 42 | 7.5 | 50 | 64 |
| 49 | 40 | 38 | 66 | 44 | 15 | 44 | 68 |
| 53 | 50 | 40 | 68 | 100 | 5 | 60 | 88 |
| 58 | 46 | 46 | 68 | 46 | 35 | 60 | 66 |
| 61 | 56 | 42 | 98 | 48 | 100 | 86 | 66 |
| 62 | 58 | 48 | 70 | 98 | 10 | 90 | 94 |
| 64 | 52 | 44 | 72 | 50 | 10 | 58 | 66 |
| 64 | 54 | 90 | 96 | 96 | 5 | 54 | 76 |
| 66 | 55 | 44 | 72 | 52 | 25 | 58 | 84 |
| 69 | 60 | 50 | 96 | 54 | 50 | 60 | 88 |
| 70 | 60 | 50 | 94 | 94 | 5 | 74 | 64 |
| 72 | 61 | 52 | 74 | 56 | 12.5 | 78 | 60 |
| 73 | 62 | 88 | 74 | 92 | 10 | 54 | 72 |
| 76 | 66 | 54 | 94 | 58 | 10 | 56 | 68 |
| 78 | 64 | 56 | 78 | 60 | 5 | 58 | 78 |
| 79 | 68 | 86 | 78 | 62 | 15 | 28 | 52 |
| 80 | 78 | 84 | 92 | 64 | 40 | 42 | 82 |
| 81 | 75 | 58 | 80 | 90 | 15 | 50 | 70 |
| 83 | 70 | 60 | 82 | 66 | 25 | 56 | 66 |
| 84 | 80 | 62 | 82 | 88 | 7.5 | 60 | 68 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 85 | 72 | 82 | 92 | 68 | 50 | 64 | 78 |
| 87 | 82 | 64 | 88 | 86 | 35 | 72 | 84 |
| 89 | 85 | 66 | 82 | 70 | 10 | 76 | 86 |
| 90 | 90 | 68 | 90 | 84 | 20 | 84 | 60 |
| 91 | 95 | 70 | 86 | 72 | 40 | 88 | 66 |
| 92 | 88 | 72 | 86 | 74 | 12.5 | 52 | 82 |
| 93 | 93 | 74 | 84 | 76 | 35 | 72 | 68 |
| 93 | 98 | 76 | 90 | 82 | 50 | 56 | 86 |
| 94 | 99 | 78 | 88 | 80 | 50 | 60 | 80 |
| 95 | 100 | 80 | 84 | 78 | 65 | 60 | 98 |

Having the full data we were able to commit correlation analysis with the main goal finding the connection between Performance and the 7 personal traits (Georgieva et al, 2010 e). This analysis can be seen below.

**Table 26 Correlation Analysis**

| | Motivation | Conscient. | Openness | Agreeab. | Experience | Extravers. | Emot.stab. |
|---|---|---|---|---|---|---|---|
| **Performance** | **0.968941** | **0.721512** | **0.598376** | **0.416717** | **0.251489** | **0.194627** | **0.128402** |

We will use these results in order to choose the factors for building our predictive mathematical model in the next chapter.

## 5.4 Summary over the definition & evaluation of the IT Human Factors

We have introduced the well-known Big Five theory in order to match the already discovered Software/IT Human Factors to the five factors and to measure them in this way. Adding two new traits to the basic ones gave us the possibility to cover the complexity of the critical human factors for the software process and to evaluate them. The factors that we have examined are listed below:

1. Openness
2. Conscientiousness
3. Extroversion
4. Agreeableness
5. Neuroticism
6. Experience
7. Motivation

and we can summarize

$$^{BigFive}HF^{FMEA}_{SoftwareProcess} : \ BigFive\,(HF^{FMEA}_{SoftwareProcess})$$

$$= \{Openness,\ Conscientiousness,\ Extroversion,\ Agreeableness,\ Neuroticism,\ Experience,\ Motivation\}$$

This transformation of role-based human factor to a list of seven characteristics is visualized in the following figure.

Having the test ready we have used an on-line platform to distribute it between different software companies and after this to evaluate the results. Analyzing them we have found the correlation between the Performance and the other seven factors and we were able to observe that the biggest correlation values are for the traits: ***Motivation; Conscientiousness; Openness and Agreeableness***.

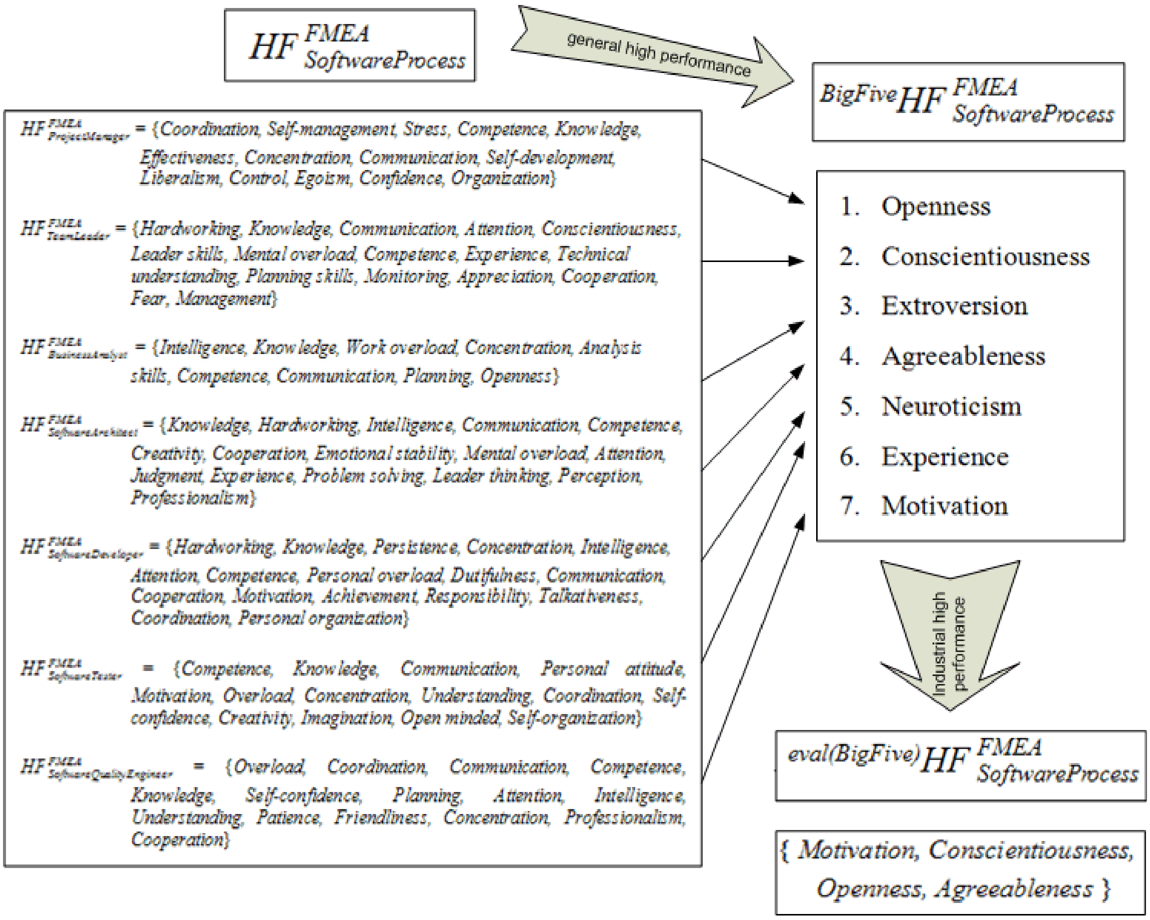



**Figure 26 Mapping of role-based human factors to Big-Five and their industrial evaluation**

That could be summarized as

$$^{eval(BigFive)}HF^{FMEA}_{SoftwareProcess} = \{\ Motivation,\ Conscientiousness,\ Openness,\ Agreeableness\ \}$$

These four personal characteristics are playing the main role in the process of model development for the IT human resources performance prediction.

# 6 Chapter – Development of the model for IT human performance prediction

The objective of the thesis is to develop a model that is able to evaluate, model and prognosticate the employees' performance. In order to achieve this we needed first to look for eventually existing solutions (Chapter 2) and to analyze the software process itself and its organization in the form of software teams (Chapter 3) with special roles. We had to analyze these roles (Chapter 4) in order to find the most important human features that influence the software process and to find a method that can describe the relationship between the already discovered software human factors and the way that they influence the employees' productiveness (Chapter 5).

As consequence we needed a method that with a defined number of trials and using the gained data from the previous chapters will give us maximum information about the mathematical dependence that we are looking for. A method that can prove that this dependence is correct and can describe it with a mathematical model. In the present chapter we will describe the development of the model for IT human performance prediction and we will end with the desired mathematical model that describes the connection between the special psychological traits and the performance. In the next chapter 7 we will experimentally prove its effectiveness and correctness.

## 6.1. Experimental design for the need of the IT human performance prediction

Looking for a method that can be applied for the development of the desired mathematical model we had to meet some restrictions:

- We had limited amount of gained data from the IT personnel.
- We had to develop the model with a minimum of experiments (because of the limited data).
- We had to find the connection between the selected personal features and the software productivity.

Having this in mind we have chosen to adopt the Design of Experiment because of the following advantages (Shivhare & McCreath 2010):

- ''gain maximum information from a specified number of experiments;
- study effects individually by varying all operating parameters simultaneously;
- take account of variability in experiments or processes themselves;
- characterize acceptable ranges of key and critical process parameters contributing to identification of a design space, which helps to provide an "assurance of quality."

We are focusing on experimentation run in the laboratory or on a piece of paper aimed at quantifying the effect of one or more variables over a certain end effect or end parameter. Thus we apply the techniques of Experimental Design and Analysis (founded over 80 years ago by Sir Ronald Fisher). The experiments supported by this technique aim to quantify the effect of qualitative variables over one particular end variable/product/effect that can be separately quantitatively measured.
We can visualize the process as a combination of different factors (controllable or not) that transform the input into some output with special characteristics.

Here is a short explanation of the steps in the chosen method (Montgomery 2008):

### *I. Recognition of and statement of the problem*

First we have to formulate the problem that we want to resolve, we have to understand its nature and to find all different factors that influence it. A clear statement of the problem often contributes substantially to better understanding of the phenomenon being studied and the final solution.

### *II. Pre-planning of the Experiment*

1. ***Choice of factors, levels and range.***
   We have to choose the input factors that we are going to analyze later, that are important for our experiment. There are different types of factors: potential design factors, held-constant factors, allowed-to-vary factors and so on but we will not discuss them because they are not concerning our particular experiment. When we are ready with the selection of the input factors, we have to decide how this factors will change, in what range and the specific levels at which runs will be made.

2. ***Selection of the response variable.*** In selecting the response variable, we should be certain that this variable really provides useful information about the process under study. In our case we do not have any doubts which is the response variable as we have a special type of passive experiments, which we will later explain and because of this we know which is our response variable and what exactly we want to observe about it.

3. ***Choice of experimental design.***
   When we have the pre-experimental planning and we are ready with our factors and response variable we have to make the next decision about the particular design. We have to consider the number of replicates, the selection of a suitable run order for the experimental trials, and the determination of whether or not blocking or other randomization restrictions are involved. Also we have to decide what type of design we are going to use for our modeling process. In our work we have chosen the central composite rotatable design, introduced by Box and Hunter (Box & Hunter 1957) (Box, Hunter & Hunter 1978) because it is the best design to build an invariant response surface. We will discuss it later.

### *III. Performing the experiment and analysis of the gained results*

4. While ***conducting the experiment***, it is vital to monitor the process carefully and to ensure that everything is being done according to the plan. Errors in experimental procedure will destroy the experimental validity.

5. ***Statistical analysis of the data.*** Statistical methods are used to analyze the data so that the results will be clear mathematical conclusions and not observations or judgments. Hypothesis testing and model adequacy checking are important analysis techniques. We will discuss the whole process of validity check later over our designed experiment.

## 6.2. Algorithm for conducting Experimental Design

### 6.2.1 Recognition of and statement of the problem

*In the present research the task is - to obtain a predictive mathematical model for the effectiveness of the software personnel, based on the individual psychometric qualities.*

Obtaining such a model is based on experimental studies, conducted according the methodology of the planned experiment and statistical analysis for its adequacy.
The experiment is a set of targeted actions, which reveal the principle of operation of the studied object (Montgomery 2008). Depending on the nature of organization and methods for obtaining the results, the experimental studies are active and passive.

*The active experiment* is applicable only for controllable objects of experimentation. The investigator himself sets the levels of factors and maintains their values in a certain stage of the experiment (Fang, Li & Sudjianto 2006).

*The passive experiment* is represented by a passive registration of output parameter values, obtained at a given combination of input parameters (factors). In this case the investigated object is observed, without interfering with the researcher in its operation (Fang, Li & Sudjianto 2006).

In our case - when investigating the effectiveness of the software personnel, depending on the individual psychometric qualities, we are using this special type of passive experiment. Types and evaluation of the psychometric qualities (characteristics) of personnel and the related efficiency of the company are determined through the collection and processing of questionnaire data. There we are observing the current state of the firma based on a fixed set of not controlled factors.

This method is used for research work over manufacturing productions and other types of companies and for processing of the experimental results is used regression analysis (Mason, Gunst & Hess 2003). The mathematical model, gained as result from the experiment is presented by a geometrical response-surface and can have the following form, for ex. for two-factor experiment (Fig.27) (Myers, Montgomery & Cook 2009) (Box & Draper 2007).

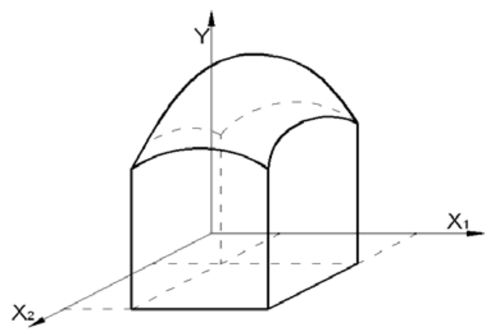



**Figure 27 Response surface for two-factor model**

If we have ***k*** factors, then the factorial space has dimension of **(*k* +1)**. When we have limited information about the objects that we are investigating, the analytical type of the response surface is unknown. Then we can assume that the surface can be represented as a part of order of Taylor in the field of experimental points of the factorial space (Atkinson & Donev 1992) and it looks like following:

$$y = b_0 + \sum_{i=1}^{k} b_i x_i + \sum_{\substack{i=1 \\ i<j}}^{k} b_{ij} x_i x_j + \sum_{\substack{i=1 \\ i<j<g}}^{k} b_{ijg} x_i x_j x_g + \ldots + \sum_{i=1}^{k} b_{ii} x_i^2 + \ldots \tag{6.1}$$

Where $y$ - is evaluation of the parameter of optimization and $x_i, x_j, \ldots, x_k$ - are coded values of the factors. $b_i, b_{ij}, b_{ijg}, \ldots, b_{ii}$ - are estimates of the regression coefficients.

Usually in the industrial practice the most commonly used models are from second degree in polynomial form, as the practice shows that in almost 100% of the cases they are adequate. (Montgomery 2008) Because of this we can reason our choice for the mathematical model of second order of Taylor and we can continue with the pre-planning of the experiment.

### 6.2.2 Pre-planning of the Experiment

The pre-planning of experiment includes all actions of preparation for conducting the planned experiment. They are as follows:

- Collection, compilation and analysis of the a priori information and conducting of preliminary single-factor experiments;
- Analysis and selection of the parameter/s of optimization. Choosing the one that most fully and accurately characterizes the object of study;
- Analysis and selection of the factors affecting the optimization parameter;
- Analysis of the factorial space; choice of domain of a function and local domain of change of the factors; determining the zero point (beginning) of the matrix of the planned experiment, the intervals of variation of the factors and the coordinates of all matrix points of the planned experiment.

6.2.2.1 Parameters of optimization and requirements to them

Optimization parameters are quantitative characteristics of the objective of study, which allow establishing of the existing relations between input and output parameters of the system. From mathematical point of view, the searching of such relations is possible only in the presence of single parameter of optimization.

The optimization parameters can vary depending on the type of the object and the purpose of the work. Conditionally we can divide them into: economical, techno-economical, technological and statistical. They must meet the following requirements (Montgomery 2008):

- The parameter of optimization must clearly, effectively and with sufficient completeness characterize the object of study;

- It must be quantitative and be assigned with a certain value;
- The requirement of uniqueness in the statistical sense is that for a set of factor values corresponds a single value of the optimization parameter;
- Under universality of the criterion of optimization must be understood its ability to comprehensively characterize the object;
- The parameter of optimization should have a clear physical sense, should be understandable for the researcher and easy to measure.

6.2.2.2 Input factors and requirements to them

The number of factors in industrial research is very large. The researcher seeks to include in the study all the relevant factors that determine the functioning of the object. To the input factors there are a number of requirements (Montgomery 2008):

- Be manageable - to accept values which are kept constant throughout the experiment, or change in some predictable way;
- Be unique - not to be a function of other factors;
- Be consistent - all combinations are feasible and safe;
- Be independent - there is no correlation between the factors. This is particularly important in the passive experiments because one factor is difficult to manage if it is a function of another;
- Have a quantitative assessment and to have a high degree of correlation with the parameter of optimization.

Each factor has its own domain of a function. The boundaries of this domain are usually set with rigid restrictions that no one can corrupt in the process of experimentation. The domain boundaries give the factor space in which to obtain an adequate mathematical model.

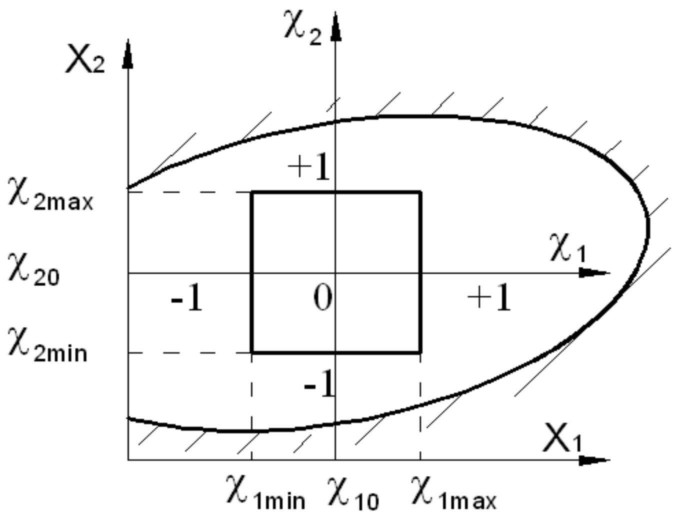



**Figure 28 Domain of a function for two-factor experiment**

After selecting the domain of a function we should find the local area for conducting the experiment. In that local area the factors change their values in the process of implementation of the planned experiment. The local area is smaller than the whole domain of the function. In general, the factors are size variables, their dimensionality can be different and also their numerical values can be of a different type. Because of this usually the experiment is not done in the original dimensions but in coded one, which is a linear translational conversion of the factorial space.

Coding is preceded by selecting the position of the center of the new coordinate system ("0" or $X_0$) and choice of the variation interval determining the location of the upper and lower limits of each factor during the experiment $Xi_{\max}$ and $Xi_{\min}$ (Fig. 28). Coding is performed by mathematical translation of the coordinate system in the new one with zero point with coordinates $X_{10}, X_{20}, \ldots, X_{k0}$ (point "0" in Figure 28).

The "0" point is called the center of the planned experiment in coded values. In the new coded space the maximum (upper) level of the factor corresponds to 1 and the minimum (lowest) to -1. The formulas for the transition (Brownlee 1965) (Cox 1957) (Davies 1967) from natural in coded values and vice versa are given below.

$$x_i = \frac{Xi - Xi_0}{\Delta Xi} \tag{6.2}$$

$$Xi = Xi_0 + \Delta Xi . x_i$$

where $\Delta Xi = \dfrac{Xi_{\max} - Xi_{\min}}{2}$

is called interval of variation (sometimes semi-interval) and $Xi$ is the coded value of the i-th factor.

Having the zero point determined from the min and max values of a factor, we should choose the variance intervals (+/-1) in a way that the values of the star points (in our case +/-1.682) are inside the factor space, otherwise our experiment will be not correct as we won't be able to cover all needed points. The particular calculations of these values are shown in 6.3.3 on Figure 38.

#### 6.2.2.3 Select the type of the planned experiment

We are choosing to use the central composite rotatable plan from the type $2^k$ because of its advantages, explained on the next page. Then we have to choose the domain of a function and the local domain for each of the factors, and we have to pay special attention when choosing the center of the experiment as it is the starting point in the planning process. For zero point is taken this point of the factorial space in which previously have been held single-factor experiments, which give information that there is expected to be localized region closest to the response optimum. The domain area must cover all points of the planned experiment, including the "star" points (explained later).

The determination of the size of the domain area is done by conducting preliminary single-factor experiments with each of the factors. The single-factor experiments indicate the type of interaction of each factor with the parameter of optimization (linear or second degree). They also show the correlation degree between each factor and the optimization parameter. The correlation degree is taken as an indicator showing which of the factors has greater (or less) influence on the optimization parameter. This is used to sort out the factors according to their influence degree, which reflects the choice of the type of the planned experiment (how many factors and how they will be included in the matrix of the experiment). The conduction of the preliminary single-factor experiments provides information about the size and range of variation for each factor and consequently about the zero-point of the plan and the value of the variance interval. It describes the size of the hypercube side (when working with coded values) in the planned experiment.

The next stage of the experimental research is to decide which of the factors will be included in the plan of the experiment. Factors by which the optimization parameter has extreme values and the correlation coefficient is high are included with priority in the matrix of the planned experiment.

### 6.2.3 Performing the experiment and analysis of the results

6.2.3.1 Planning of the experiment

In planning of the experiment is included: determining the plan of the experiment; determining the necessary and sufficient number of experiments and observations with the already chosen model of design; establishing the matrix of the experiments and randomization of the trials.

The plan of this experiment is a set of data - specifying the number, the conditions and the sequence of implementation of necessary and sufficient trials in order to solve the task with the needed accuracy. It is presented in the form of a design matrix (rectangular table), the rows of which satisfy the tests and their position in the factor space, and the columns - the coded values of the factors and the parameter of optimization (Table 28).

The analysis of the gained results includes the calculation and statistical estimation of the coefficients of the model; writing the gained mathematical model in coded and natural values and examining its adequacy.

The type of the mathematical model, whose coefficients we will determine (calculate) is as already explained chosen to be from second degree and it determines the structure of the planned experiment. It will also be of second degree, this means that it will consist from experimental points at the end-points of the cube (hypercube); it can have two, three or more changeable factors; it will have duplicated experimental points in the center of the plan and two "star" points (explained later) for each axis of the factorial space.

There are a lot of possibilities to realize the matrix of the planned experiment (central composite orthogonal design; central composite rotatable design; "D"-optimal plans, plans of Hartly, etc..) but we choose to work with the central composite rotatable design for its advantages. This method is proposed by Box and Hunter (Box & Hunter 1957) and (Cohran & Cox 1957) and later examined by Myers (Myers 1971).

It offers the following advantages (Khuri & Cornell 1996) (Myers, Montgomery & Cook 2009):

- Ensures the invariance of the plan and of the parameter of optimization by rotating the coordinate system around its center;
- The model obtained by the rotatable plan describes the response surface with equal accuracy (equal variance) in all directions of the coordinate axes;
- Surface lines of the same value of variance are concentric circles or hyper-spheres with a center coinciding with the beginning of the coordinate system;
- The variances of the mathematical model are the same for all points that are equidistant from the design center and have the minimum values;

*Central composite rotatable plan is built* (Montgomery 2008) using the following common construction rules:

- Build a full factorial experiment with a number of experiments $N_1 = 2^k$;
- To the experimental points of the full factorial experiment are added experiments in 2k "star points" located at a distance of ± α (star arm) from the center of the plan; the values of α are calculated according to formula (6.3).
- To all these experimental points are added $N_0$ observations in the center of the plan $(x_i = 0)$;
- "k" is the number of the changing factors.

The difference between central composite rotatable and central composite orthogonal plans lies in the manner of selecting the size values of the star arm α and the number of observations in the center of the plan. The size of the star arm by central composite rotatable plan is calculated based on the condition of invariance of the plan. This calculation is done by the formula:

$$\alpha = \sqrt[4]{N_1} = 2^{\frac{k}{4}} \tag{6.3}$$

The number of the duplicate observations $N_0$ in the center of the plan is chosen so as to achieve uniformity. This means that we should obtain almost identical values of dispersion (variance) of the optimization parameter in the factor space and the number of observations should also be sufficient for statistical analysis of the results.

The planning, where through suitable choice of the number of observations in the center of the plan can be achieved almost equal distribution of the variance in the whole area and the variance has the same value for all equidistant from the center points is called *rotatable-uniform planning.*

- To provide uniformity of the plan, $N_0$ is determined by the relationship (Dean & Voss 1999)

$$N_0 = \lambda(N_1 + 4\sqrt{N_1 + 4}) - N_1 - 2k \tag{6.4}$$

Where λ = 0,7844 ; 0,8385 ; 0,8705 ; 0,8918 ; 0,907 ; 0,9185 and k = 2,3,4,5,6,7. (Dean & Voss 1999)

- To assure an orthogonal rotatable plan, $N_0$ is determined by the relationship (Dean & Voss 1999)

$$N_0 = 4\sqrt{N_1} - 2k + 4 \tag{6.5}$$

The necessary data to build a central composite rotatable plan can be seen from table 27 and table 28 shows the data for central composite rotatable plan with k = 3.

**Table 27 Number of experimental points and size of the star arm by rotatable plans with different numbers of factors (Dean & Voss 1999)**

| $k$ | $N_1$ | $N_\alpha$ | $N_0$ | $N$ | $\alpha$ |
|---|---|---|---|---|---|
| 2 | $2^2$ | 4 | 5 | 13 | 1,414 |
| 3 | $2^3$ | 6 | 6 | 20 | 1,682 |
| 4 | $2^4$ | 8 | 7 | 31 | 2,000 |
| 5 | $2^5$ | 10 | 10 | 52 | 2,378 |
| 6 | $2^6$ | 12 | 15 | 91 | 2,828 |
| 7 | $2^7$ | 14 | 21 | 163 | 3,333 |

**Table 28 Matrix for rotatable plan of second level - type $2^3$ (factors are in coded form)**

| | Nr of the experiment | $X_1$ | $X_2$ | $X_3$ | $y$ |
|---|---|---|---|---|---|
| Full factorial experiment $2^3$ | 1 | -1 | -1 | -1 | $y_1$ |
| | 2 | +1 | -1 | -1 | $y_2$ |
| | 3 | -1 | +1 | -1 | $y_3$ |
| | 4 | +1 | +1 | -1 | $y_4$ |
| | 5 | -1 | -1 | +1 | $y_5$ |
| | 6 | +1 | -1 | +1 | $y_6$ |
| | 7 | -1 | +1 | +1 | $y_7$ |
| | 8 | +1 | +1 | +1 | $y_8$ |
| ''Star'' points | 9 | -1,682 | 0 | 0 | $y_9$ |
| | 10 | +1,682 | 0 | 0 | $y_{10}$ |
| | 11 | 0 | -1,682 | 0 | $y_{11}$ |
| | 12 | 0 | +1,682 | 0 | $y_{12}$ |
| | 13 | 0 | 0 | -1,682 | $y_{13}$ |
| | 14 | 0 | 0 | +1,682 | $y_{14}$ |
| Experiments in the center of the plan | 15 | 0 | 0 | 0 | $y_{15}$ |
| | 16 | 0 | 0 | 0 | $y_{16}$ |
| | 17 | 0 | 0 | 0 | $y_{17}$ |
| | 18 | 0 | 0 | 0 | $y_{18}$ |
| | 19 | 0 | 0 | 0 | $y_{19}$ |
| | 20 | 0 | 0 | 0 | $y_{20}$ |

By equal number of observations in the experimental points, the estimates of the coefficients in the regression equation are determined by the dependencies: (Dean & Voss 1999)

$$b_0 = a_1 \sum_{l=1}^{N} y_l - a_2 \sum_{i=1}^{k} \sum_{l=1}^{N} x_{il}^2 y_l$$

$$b_i = a_3 \sum_{l=1}^{N} x_{il} y_l$$

$$b_{ij} = a_4 \sum_{l=1}^{N} x_{il} x_{jl} y_l$$

$$b_{ii} = a_5 \sum_{l=1}^{N} x_{il}^2 y_l + a_6 \sum_{i=1}^{k} \sum_{l=1}^{N} x_{il}^2 y_l - a_2 \sum_{l=1}^{N} y_l \tag{6.6}$$

Where $a_1, a_2, a_3, a_4, a_5, a_6$ are defined from the following table 29 depending on the number of factors and the type of the plan. Values of the coefficients $a$ are used to calculate the estimates of the coefficients $b$ in the regression equations obtained with central composite orthogonal plan or central composite rotatable plan.

**Table 29 Values of the coefficients $a$ (Dean & Voss 1999)**

| $k$ | $N_1$ | Central Composite Rotatable Plan | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ |
| 2 | $2^2$ | 0,2000 | 0,1000 | 0,1250 | 0,2500 | 0,1250 | 0,0187 | 0,1000 |
| 3 | $2^3$ | 0,1663 | 0,0568 | 0,0732 | 0,1250 | 0,0625 | 0,0069 | 0,0568 |
| 4 | $2^4$ | 0,1429 | 0,0357 | 0,0417 | 0,0625 | 0,0312 | 0,0037 | 0,0357 |

The estimates of the variances of all the regression coefficients are calculated by the formulas: (Dean & Voss 1999)

$$s^2[b_0] = a_1 s^2[y];$$

$$s^2[b_i] = a_3 s^2[y];$$

$$s^2[b_{ij}] = a_4 s^2[y];$$

$$s^2[b_{ii}] = a_7 s^2[y]. \tag{6.7}$$

Where $s^2_{repro}[y] = \frac{1}{N_0 - 1} \sum_{u=1}^{N_0} (y_{0u} - y_{0mean})^2$ is dispersion of reproducibility. (6.8)

$N_0$ is the number of experiments in the '0'-point. $y_{0u}$ are the real values of $y$ in the '0'-point and $y_{0mean}$ is their mean value.

6.2.3.2 Statistical analysis

If $\left|b_{0,i,ij,ii}\right| > s\left[b_{0,i,ij,ii}\right]t_{Student}$, where $t_{Student}$ is the 'coefficient of *Student*' - a table value (Dean & Voss 1999), then the corresponding coefficient is significant. This means that the coefficient is important for the regression equation and will take part in it. In this manner are checked the conditions of importance for all regression coefficients.

*The validation for adequacy of the whole mathematical model is done by the criterion of Fisher:*

$$F_{calculated} = \frac{s^2_{adequacy}}{s^2_{reproducibility}} \qquad \text{(Fisher \& Yates 1973) (Dean \& Voss 1999)} \qquad (6.9)$$

Where the dispersion (variance) of adequacy is determined by the dependence:

$$s^2_{adequacy} = \frac{\sum_{l=1}^{N=20}\left(y_{measured} - y_{calculated}\right)^2}{N - k' - \left(N_0 - 1\right)} \qquad \text{(Dean \& Voss 1999)} \qquad (6.10)$$

$N$ is the number of the experiments, $N_0$ is the number of experiments in the '0'-point and $k'$ is the number of significant coefficients in the gained mathematical model.

If $F_{calculated} \le F_{table}$ the model is adequate, this means that it is true and gives exact results but if $F_{calculated} > F_{table}$ the model is not adequate and we cannot use it because it does not describe the experiment in a useful manner. The values of $F_{table}$ are taken from (Dean & Voss 1999).

6.2.3.3 Interpretation of the results

The planning of experiment from second order finishes with finding an adequate quadratic equation (mathematical model) of the form (Dean & Voss 1999):

$$y = b_0 + \sum_{i=1}^{k} b_i x_i + \sum_{\substack{i=1 \\ i<j}}^{k} b_{ij} x_i x_j + \sum_{\substack{i=1 \\ i<j<g}}^{k} b_{ijg} x_i x_j x_g + \ldots + \sum_{i=1}^{k} b_{ii} x_i^2 + \ldots \qquad (6.11)$$

The obtained model must be analyzed in order to find the nature of the response surface in the examined area. There should also be checked if there exists a maximum point (extremum) for this surface and if so, then the coordinates of this point have to be found.

This analysis begins with the transformation of the above equation into canonical form. The canonical transformation is presented by the choice of a new coordinate system, which greatly facilitates the geometric analysis of the equation. This transformation is expressed in determining the center of the response surface (if it exists), then relocating the original coordinate center into the found new one (by this relocation the linear members $b_i x_i$ are dropped out) and after this rotating the coordinate axes (by this rotation the members $b_i x_i x_j$ are dropped out too). Having all these changes, the quadratic equation of the response surface in canonical form looks like this:

$$y = y_s + \theta_{11} z_1^2 + \theta_{22} z_2^2 + ... + \theta_{mm} z_m^2 \quad \text{(Dean \& Voss 1999)} \qquad (6.12)$$

Where $y_s$ is the value of the response surface in the center of the new coordinate system; $z_i$ - are the new coordinate axes rotated in the factor space with a special angle to the old ones Xi; $\theta_{ii}$ - are the canonical coefficients.

*The procedure for the canonical transformation of the model contains the following steps:* (Myers, Montgomery & Cook 2009)

1. Determine the coordinates of the center of the response surface ($x_{1s}, x_{2s}, ... x_{is} ... x_{ms}$) by solving the system of linear equations, obtained after aligning to zero the first derivative of $y$ for each $x_i$;

$$\frac{\partial y}{\partial x_i} = 0, \; i = 1,...,k \qquad (6.13)$$

If the determinant of system (6.13) is not equal to zero, the response surface has a center, but if it is equal to zero then the surface does not have a center within the factorial space. In this case the center is accepted to be either in the beginning of the old coordinate system or in a point that holds the "best" response value.

2. Calculating the surface response value in the new center - $y_s$ (or finding the free member of the canonical equation). This is done as the already calculated coordinates $x_{is}$ from (6.13) are substituted in equation (6.11).

3. Determination of the canonical coefficients $\theta_{ii}$. For this purpose we build the characteristic equation:

$$f(\theta) = \begin{vmatrix} b_{11} - \theta & 0{,}5b_{12} & ... & 0{,}5b_{1i} & ... & 0{,}5b_{1m} \\ 0{,}5b_{21} & b_{22-\theta} & ... & 0{,}5b_{2i} & ... & 0{,}5b_{2m} \\ ... & ... & ... & ... & ... & ... \\ 0{,}5b_{i1} & 0{,}5b_{i2} & ... & b_{ii} - \theta & ... & 0{,}5b_{im} \\ ... & ... & ... & ... & ... & ... \\ 0{,}5b_{m1} & 0{,}5b_{m2} & ... & 0{,}5b_{mi} & ... & b_{mm} - \theta \end{vmatrix} = 0 \qquad (6.14)$$

where $b_{ij} = b_{ji}$.

The canonical coefficients are roots of equation (6.14). The check up for correctness of the calculations is done by the formula:

$$\sum_{i=1}^{m} b_{ii} = \sum_{i=1}^{m} \theta_{ii} \tag{6.15}$$

4. Writing down equation (6.11) in canonical form

$$y = y_s + \theta_{11} z_1^2 + \theta_{22} z_2^2 + \ldots + \theta_{mm} z_m^2 \tag{6.16}$$

and determining the type (as geometrical figure) of the response surface.

5. Obtaining a system of equations that links the new coordinate axes with the old ones:

$$z_1 = \cos\alpha_1 (x_1 - x_{1s}) + \cos\beta_1 (x_2 - x_{2s}) + \ldots + \cos\nu_1 (x_i - x_{is}) + \ldots + \cos\omega_1 (x_m - x_{ms})$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$z_m = \cos\alpha_m (x_1 - x_{1s}) + \cos\beta_m (x_2 - x_{2s}) + \ldots + \cos\nu_m (x_i - x_{is}) + \ldots + \cos\omega_m (x_m - x_{ms}) \tag{6.17}$$

Using special formulas that we are not further examining here (Dean & Voss 1999) we can find the connection between the old and the new coordinate systems.

According to the obtained $\theta_{ii}$ values there exist different possibilities for the response surface (Myers, Montgomery & Cook 2009). This is automatically done later in the used software. Because of this here are not given any more details about the response surfaces. In the next part follows the explanation of our particular design of experiment and of the obtained results.

## 6.3 The development of the model for IT human performance prediction

### 6.3.1 Recognition and statement of the problem

The question that we have to answer, as already explained in the beginning of this chapter is how and which human factors influence the individual performance in a software company during the software development process.

In our case - when investigating the effectiveness of a software company, depending on individual psychometric qualities of the personnel, we are using this special type of passive experiment (that we have already explained above). Types and evaluation of the psychometric qualities (characteristics) of personnel and the related efficiency of the company are determined through the collection and processing of questionnaire data. There we are observing the current state of the firma based on a fixed set of not controlled factors.
As already explained in the previous chapters 3, 4 and 5 we have conducted a full examination of the software development process with all stages and with the corresponding different roles and their responsibilities. We have adopted the FMEA method to make this

analysis in order to find the most important human characteristics and then adopting the Big Five theory we were able to conduct an evaluation of the data.

After summarizing the data we have analyzed how each of the factors influences the productivity and we have calculated the needed correlation values. These correlation values are actually our analysis, which of the factors are the most important for the productiveness. From the data shown on the figures below we can also see the min and max for each of the factors, which is very important when we want to find the factor space of our experiment.

This information is shown here once again for better understanding.

**Table 30 Correlation Analysis between the personal features and the performance**

| | Motivation | Conscient. | Openness | Agreeab. | Experience | Extravers. | Emot.stab. |
|---|---|---|---|---|---|---|---|
| **Performance** | **0.968941** | **0.721512** | **0.598376** | **0.416717** | **0.251489** | **0.194627** | **0.128402** |

We can see the correlation values for the factors:

***Correlation (Motivation, Performance) = 0.96***
***Correlation (Conscientiousness, Performance) = 0.72***
***Correlation (Openness, Performance) = 0.59***
***Correlation (Agreeableness, Performance) = 0.41***
Correlation (Experience, Performance) = 0.25
Correlation (Extraversion, Performance) = 0.19
Correlation (Emotional Stability, Performance) = 0.128

Led by these results and the knowledge that correlation values between 0.3 and 0.5 have medium importance and bigger than 0.5 have big importance (Cohen 1988), it was easy to decide that we will consider the first four factors.

In order to obtain a clear idea how exactly these features influence the performance here follow the figures showing these dependencies. In all figures Series 1 are the real points and Poly. (Series 1) are the polynomial functions that are maximal near to the real values.

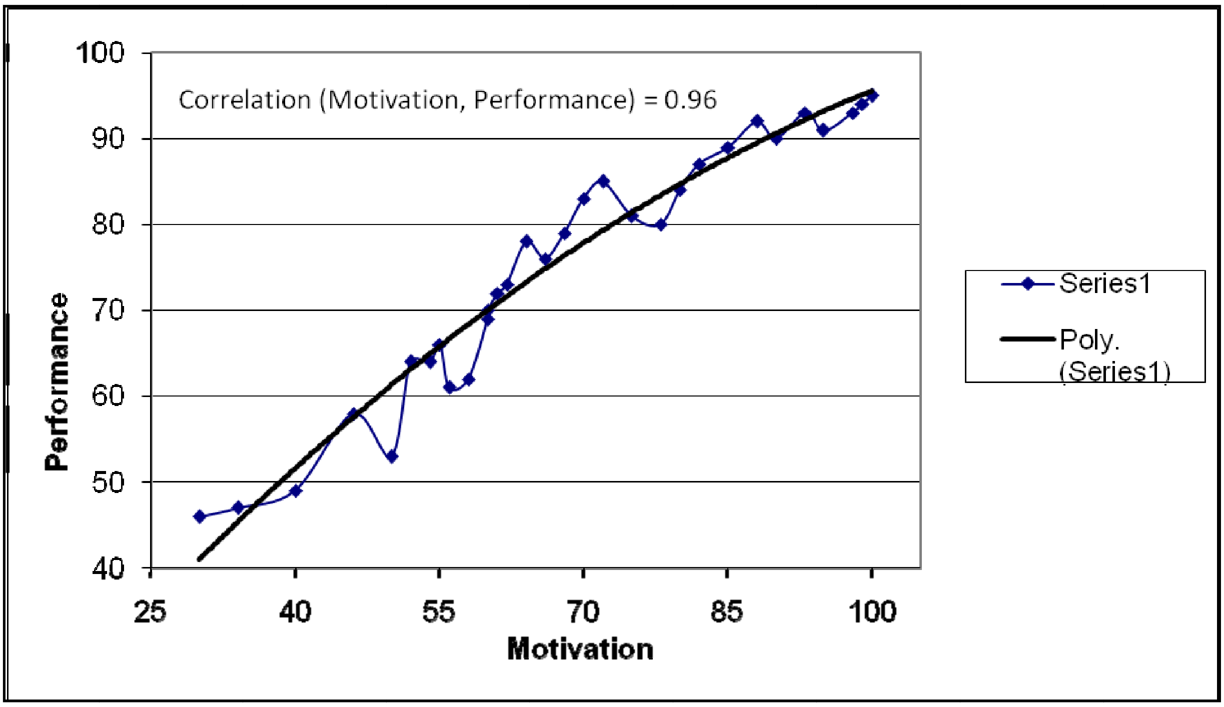



**Figure 29 Correlation between Motivation and Performance**

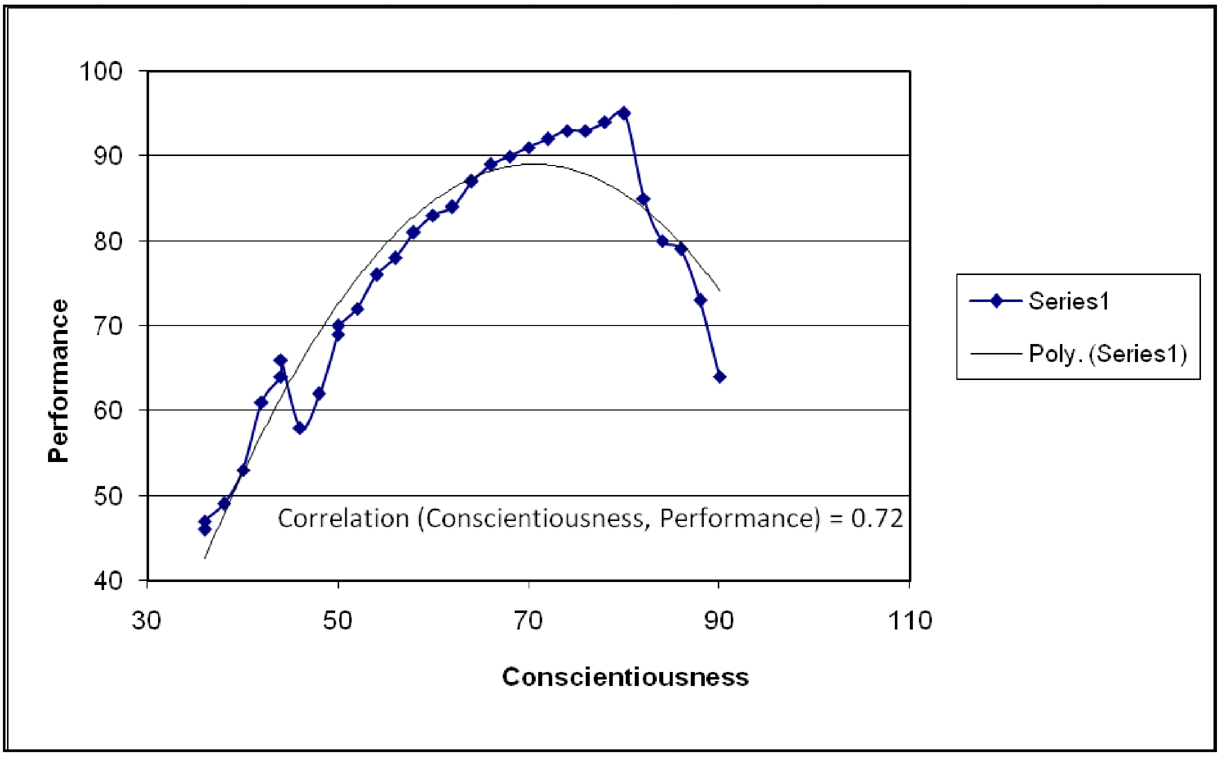



**Figure 30 Correlation between Conscientiousness and Performance**

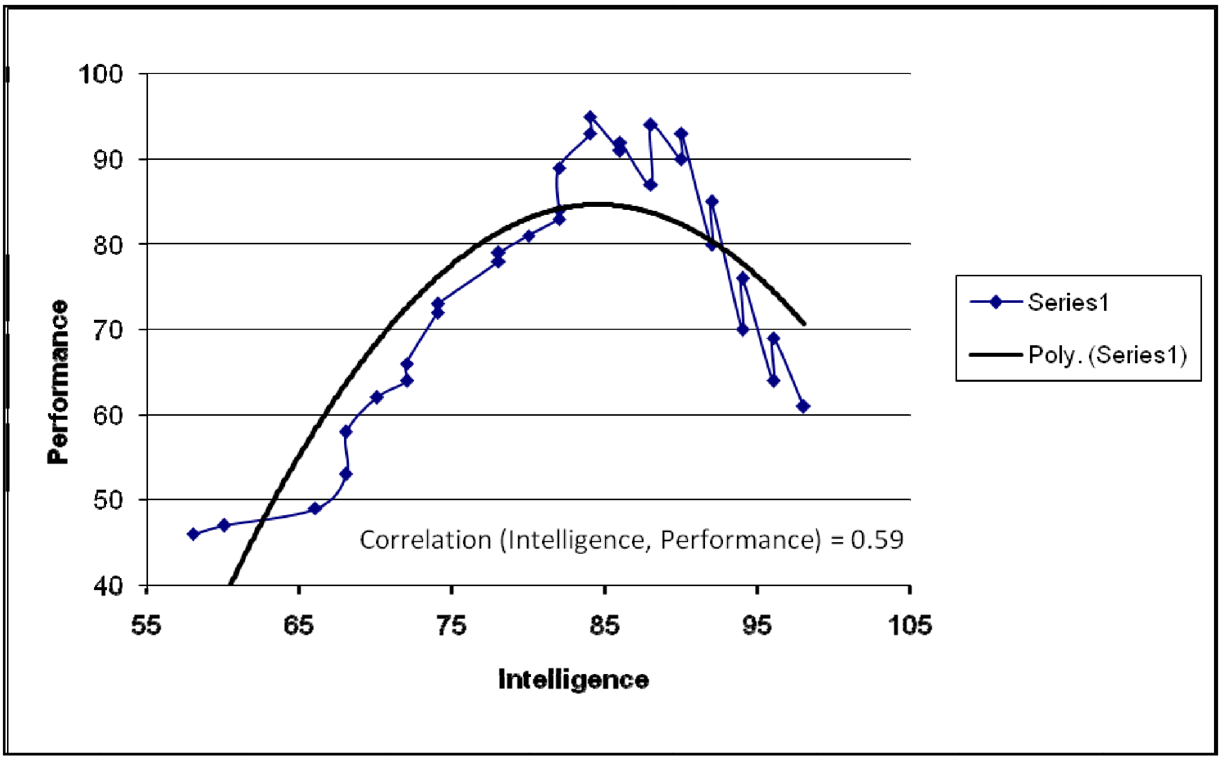



**Figure 31 Correlation between Intelligence and Performance**

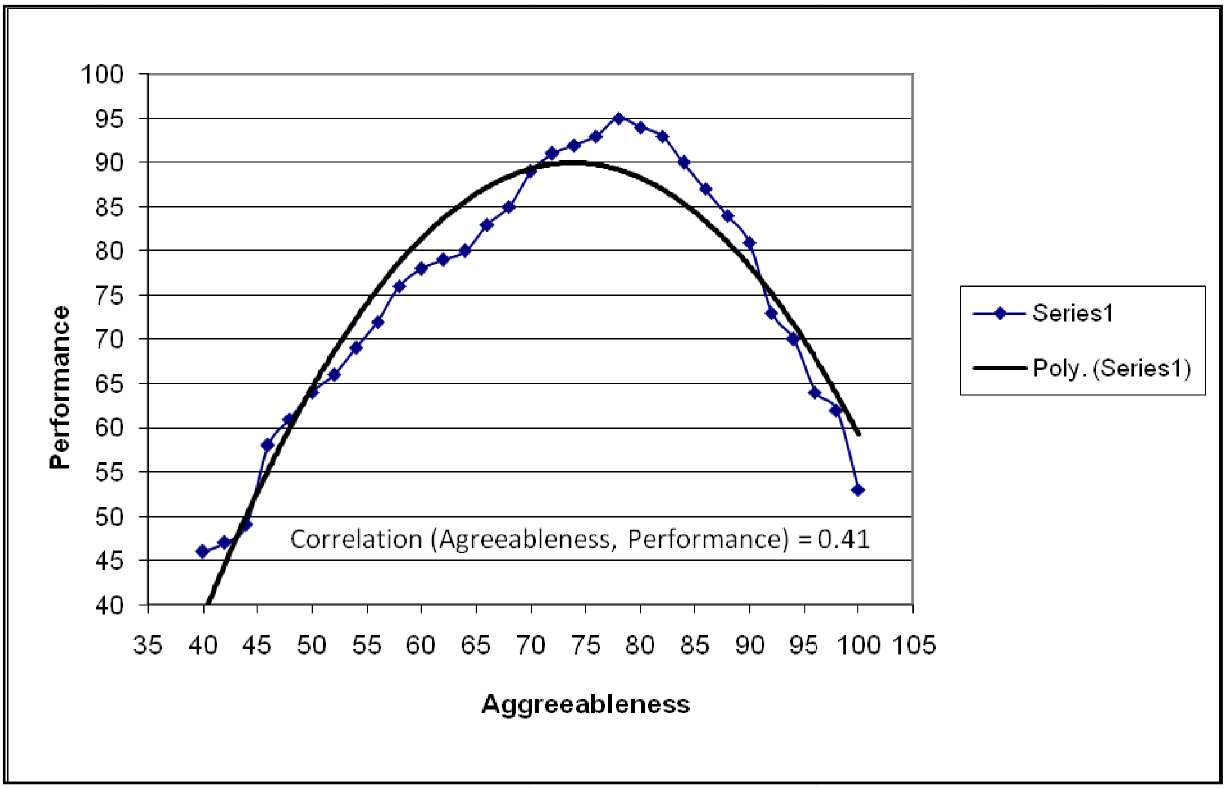



**Figure 32 Correlation between Agreeableness and Performance**

Observing the figures 30 to 32 we see parabola-like graphics and maximum of the performance values for some mean to high values of the factors. This analysis will be observed into more details in the next part of the experiment as these connections between the factors and the productiveness are actually our goal.

We can see that the graphic in figure 29 is different from the other ones. It is very near to a line and until 85% (where the maximum is) it is growing and after this we have a slow reduction in the values. This can be explained with a complex subjective-psychological dependence: the growing of the motivation up to 85% is connected with growing of the desire to give the best possible productiveness at work and by higher values than 85% this desire is decreasing. This is explained with the fact that the people with 100% of motivation find it difficult to see perspective for development as they have already reached the maximum; this is a kind of de-motivation and results in lower performance levels.

In order to make our experiment more comprehensive we have decided to make three experiments for three crucial values of the motivation: 55%; 70% and 85%. By these levels of motivation we can see a significant change in the performance values. Conducting these three experiments we can show how does the productiveness change by different values of motivation and make a comparison between them.

Processing the gathered data we have also build the correlation graphics between the other three factors (Experience, Extraversion and Emotional Stability) and the performance – Figures 33 to 35. We have calculated also the correlation values and as shown in table 30, they have significantly low values. This shows that we cannot use them as predictors of human productivity because the connection is not clear enough and because of this these factors will not be of our interest any more.

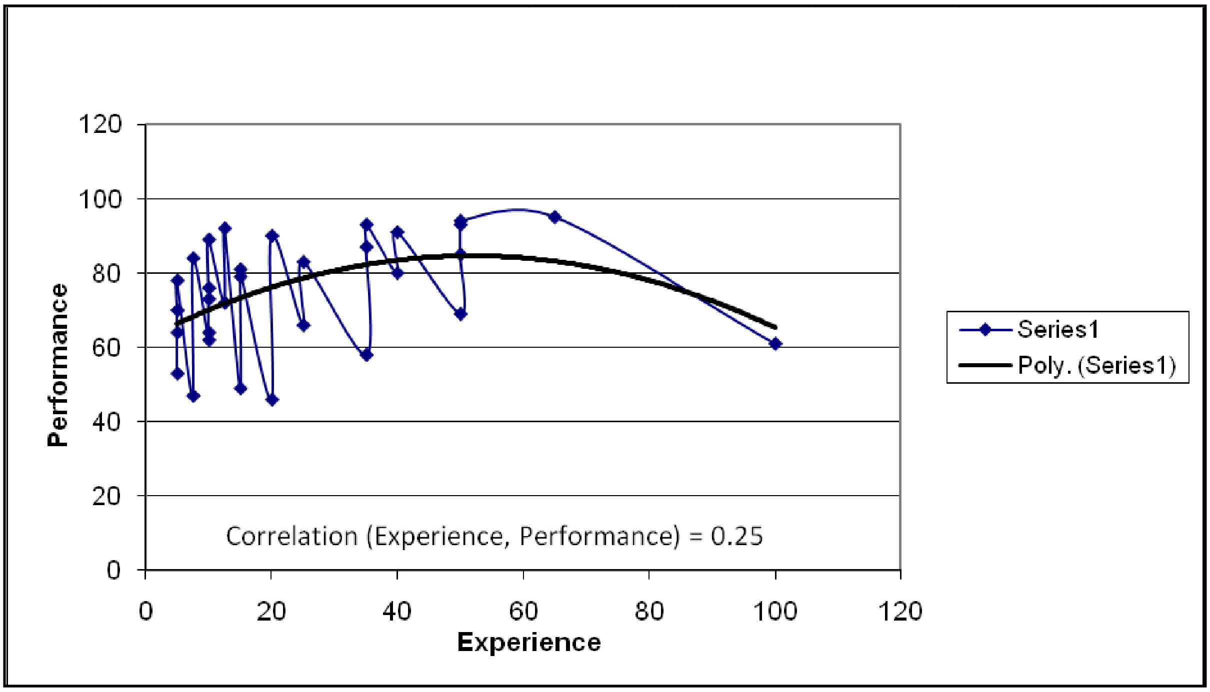



**Figure 33 Correlation between Experience and Performance**

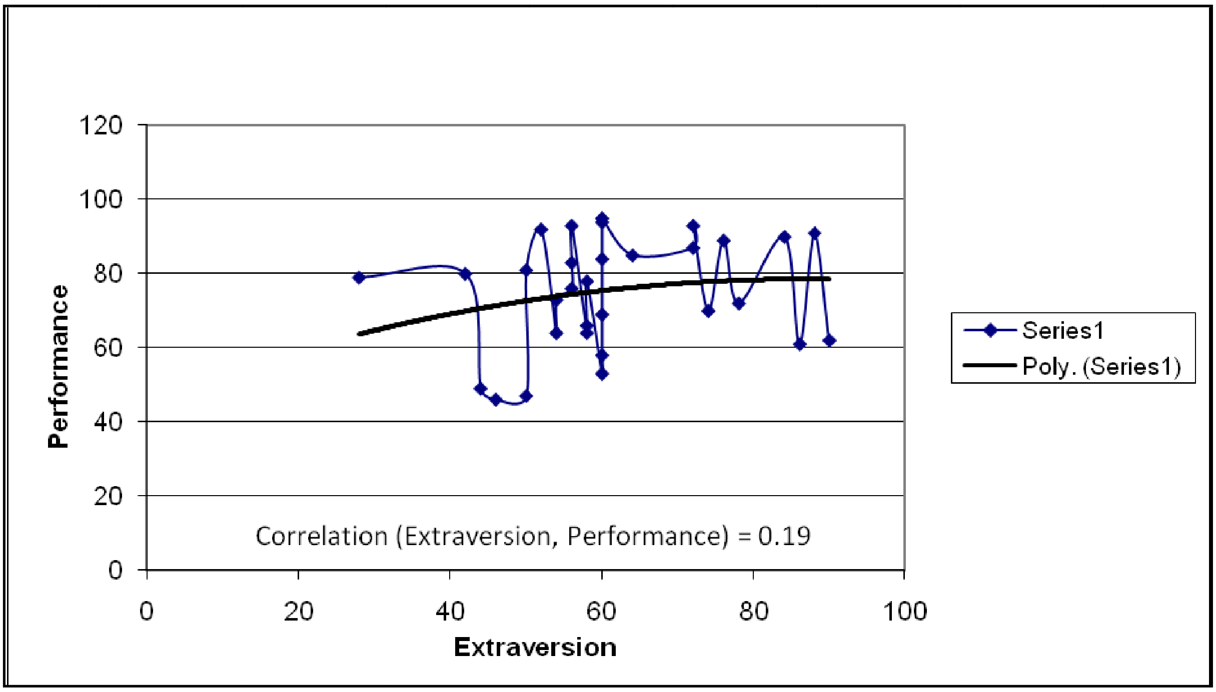



**Figure 34 Correlation between Extraversion and Performance**

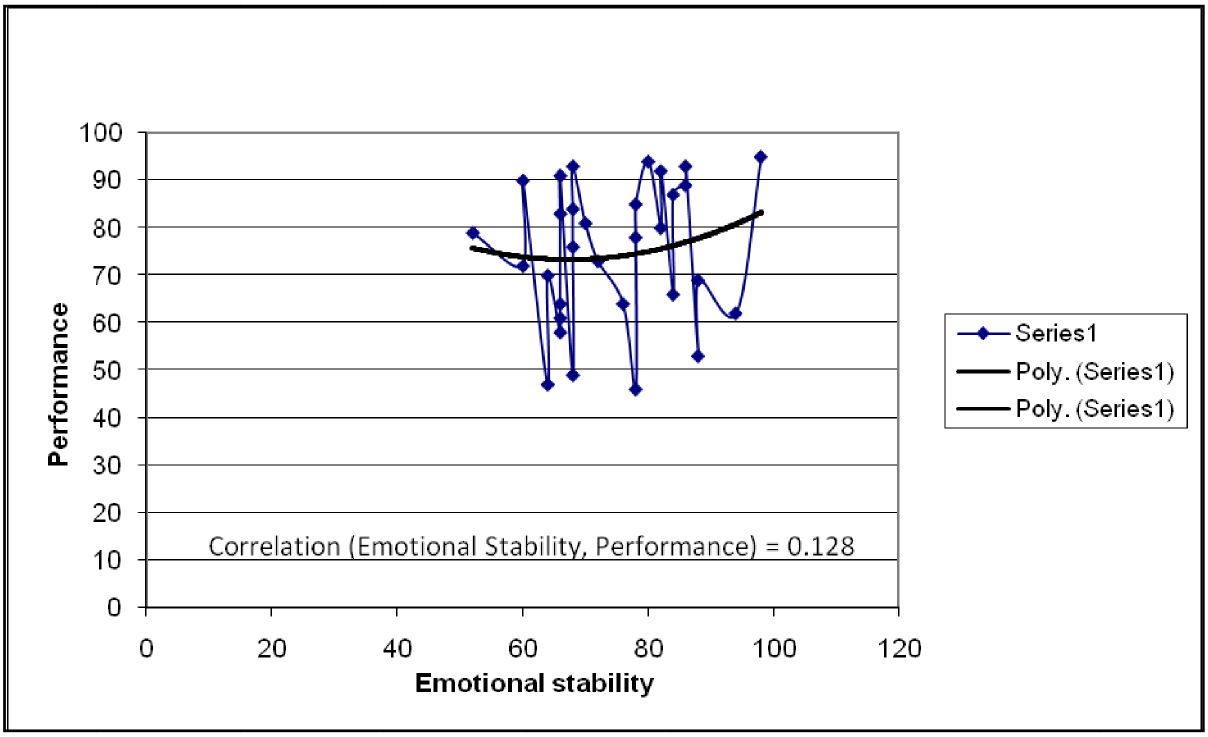



**Figure 35 Correlation between Emotional Stability and Performance**

This means that we have found the four most important human traits (***Motivation, Conscientiousness, Openness*** and ***Agreeableness***) that influence the performance and we will continue our research-work with them. From the figures we see the min and max values, which are important for determination of the factor space, and they are:

Conscientiousness from 38 % to 90%.
Intelligence from 58% to 98%.
Agreeableness from 40% to 100%.

### 6.3.2 Pre-planning of the Experiment

6.3.2.1 Choice of factors, levels and range

Having the analysis from the previous point and the theoretical background that explains how we select our input factors we can say that we have already found the factors that we will analyze and they are as follows: Motivation; Conscientiousness; Intelligence and Agreeableness.

In the industrial practice the most commonly used models are of second degree in a polynomial form. Because of this we can reason our choice for the mathematical model of second order of Taylor.

The range or the factor space in our case is determined of the values of the factors, measured during the test. And they vary in the following manner:

Conscientiousness from 38 % to 90%.
Intelligence from 58% to 98%.
Agreeableness from 40% to 100%.

As we have already explained we will conduct three experiments by motivation of 85%, 70% and 55% and this means that we have three factors which we will observe in connection with the performance. We will design an experiment for three factors at two levels, from the type $2^3$ in the already mentioned ranges but we will design three different experiments for every special value of the motivation factor.

6.3.2.2 Selection of the response variable

In our research the examined response variable is the human performance in the software development in connection with the special personal traits (*Motivation, Conscientiousness, Intelligence* and *Agreeableness*). In order to measure this performance we have decided to use the mean value between three different evaluations; the first one is the employee's personal work-evaluation; the second one is the evaluation from their colleagues and the third one is the supervisors' evaluation. In this way we have received a comprehensive value that we can use in our further experiment.

The formulas for these evaluations can be seen in chapter 3, but we will show them here once again (see (3.47) and (3.48)).

$$humanPerformance = \{HF_{IT} \times softwareDevelopmentProcess\} \quad (6.18)$$

$$humanPerformanceEvaluation = \{personalAssessment, supervisorAssessment, colleagueAssessment\}$$

$$r_{SR}^{(personalAssessment)} \in R_{SR}: personIT \times assessment \times workingProcess \rightarrow personalAssessment$$

$$r_{SR}^{(supervisorAssessment)} \in R_{SR}: personIT \times supervisor \times assessment \times workingProcess \rightarrow supervisorAssessment$$

$$r_{SR}^{(colleagueAssessment)} \in R_{SR}: personIT \times colleague \times assessment \times workingProcess \rightarrow colleagueAssessment$$

$$person_{IT} = \{analyst, designer, developer, acquisitor, reviewer, programmer, tester, administrator, qualityEngineer, project\ leader, systemProgrammer, chiefProgrammer\}$$

Furthermore, our experiment leeds to

$$HF_{IT} \rightarrow {}^{eval(BigFive)}_{DoE} HF^{FMEA}_{SoftwareProcess} \quad (6.19)$$

where *DoE* stands for the applied statistical method as so-called *Design of Experiment*.

#### 6.3.2.3 Choice of experimental design

There are a lot of possibilities to realize the matrix of the planned experiment (central composite orthogonal design; central composite rotatable design; "D"-optimal plans, plans of Hartly, etc..) but we have chosen to work with the central composite rotatable design for its advantages. This method is proposed by Box and Hunter (Box & Hunter 1957) and (Cohran & Cox 1957) and later examined by Myers (Myers 1971).

It offers the following advantages (Khuri & Cornell 1996) (Myers, Montgomery & Cook 2009):

- Guarantees the invariance of the plan and of the parameter of optimization;
- The model obtained by the rotatable plan describes the response surface with equal accuracy (equal variance) in all directions of the coordinate axes;
- In the whole factors space, the parameter of optimization has the same variance. This assures that the calculation accuracy of the optimization parameter is independent from the place where we are going to build the experiment;
- The variance of the optimization parameter does not change by rotation and translation of it coordination system. This allows us to conduct the canonical experiment (rotation

and translation of the coordination system) with the goal to find the geometrical figure of the designed experiment;

- The fact that by rotatable experiments the variance does not change assures the correctness of the statistical analysis of the gained mathematical model.

### 6.3.3 Realizing and Analysis of the Experiment

For conducting our experiments we have worked in cooperation with the Technical University of Varna, Bulgaria and have used the kindly provided from them software-tool to conduct all calculations needed for our design. In the following part we will explain in detail the used software and the results that it provides on every step. The software is in Bulgarian language developed and because of this the text in the windows is in Bulgarian, but an explanation in English assures the understanding. The summarized results of all experiments can be found in the following paper (Georgieva et al, 2011 b).

Of course it is possible to use also other software but we have two very fundamental reasons to choose exactly this one:

1. Choosing to model our experiment with the central composite rotatable design it was impossible for us to find well known software that supports us exactly with the desired steps for conducting the experiment.
2. Because of our cooperative work with the Technical University of Varna we did not have to pay for the software (whereas all other software needed to be paid for) and gained exactly the appropriate tool for our goal.

Here we are explaining the steps in conducting the experiment and we have visualized the whole process in the following screenshots. The software tool is specially developed for central composite rotatable plans and all calculations and statistical verifications are included. This makes the planning of our experiment much easier and supplies us at the end with the desired mathematical model and the response surface graphics.

The first step is to choose the number of factors (Fig.36) that we are going to include in our experiment and as already explained we have decided that we will explore three factors, so we are choosing here the second option which is ‘three factors experiment’.

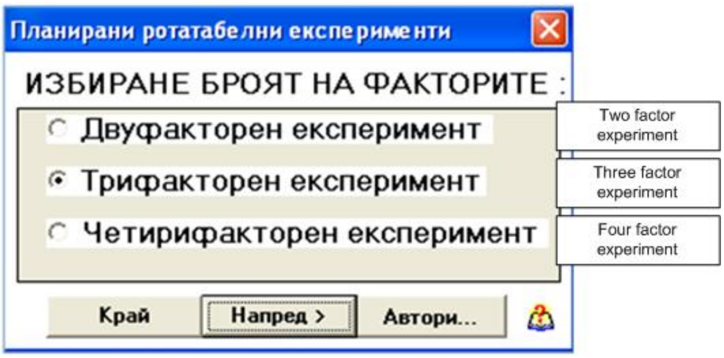



**Figure 36 Choosing the number of the factors**

On the next step (Fig.37) we have to choose names for the factors. For convenience in the software product we have decided to use the following abbreviations: *co*=conscientiousness; *int*=intelligence; *agr*=agreeableness and *pr*=performance. The dimensions for the factors according to our methodology (test data) are in percentage.

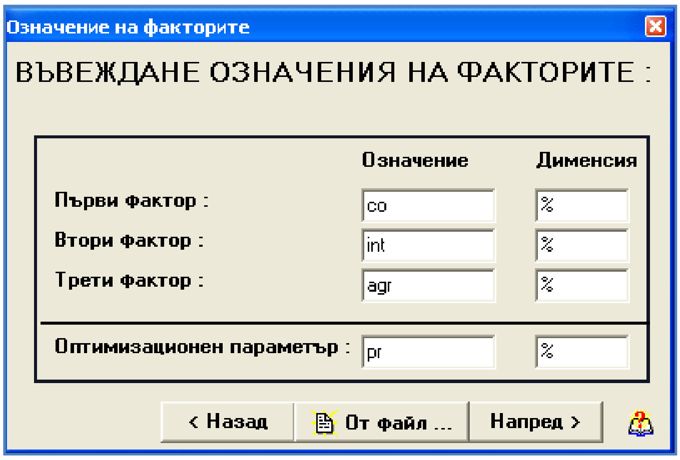



**Figure 37 Selecting names and dimensions for the factors**

On the next Figure 38 are visualized the different values of a factor that are important for the correct design of the experiment.

The first important point from the realizing of the experiment is the determination of the zero point of the coordinate system. Symmetrically around it will be build the plan of the experiment. The zero point determination is made using the values of the other factors, which can be seen on the Figure 30 to Figure 32, where are shown the correlations between the factors and the performance. Usually for zero point is chosen this one, that is in the middle of the factor space, determined by the values of the input factors.

We are going to explain here how we determine the zero-levels and the variance intervals for every factor. For the first factor conscientiousness the factor space is between 38% and 90%, this means that the middle is 64% and this will be the zero point for cons. The variance intervals (+/-1) should be chosen in a way that the values of the star points are inside the factor space, otherwise our experiment will be not correct as we won’t be able to cover all needed points. The star points are usually chosen to be on a distance from the end of the factor space not bigger than 15% of the distance between the two star-points. This is made with few experiments on the trial-error principle until the best values are found. For the conscientiousness we have the following values (see the figure), we are choosing the values of the star points so that we will be sure that they stay in the factors space after designing the experiment and these are 44% for -1.682 and 84% for +1.682 and based on them we are calculating the values of +1 and -1 (52%, 76%). This means that for the conscientiousness the variance interval will be +/- 12.

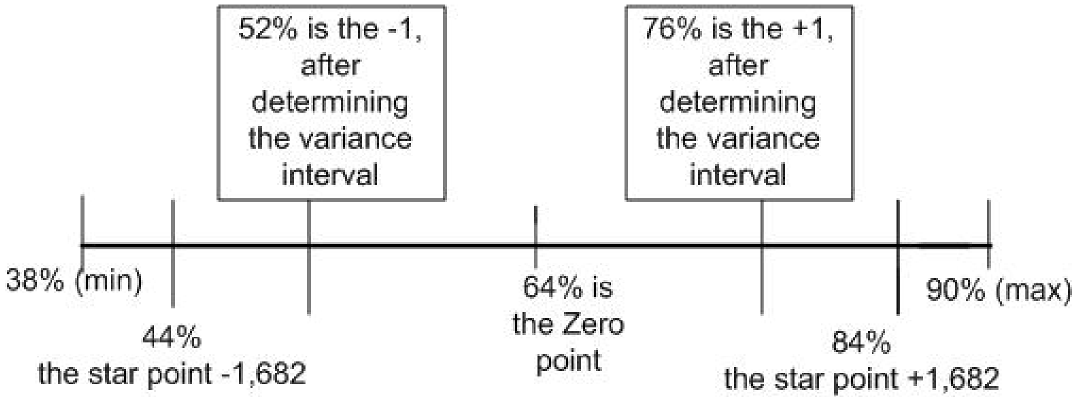



**Figure 38 Determining the input information for the conscientiousness**

Analogically we are making the same calculations for the other two factors and we gain the values:

*Intelligence* from 58% to 98%, this means that the zero point will be 78%; the star points will be approximately -1.682 = 62% and +1.682 = 94%. Then we can calculate the variance interval which will be: +/-10 and the -1 = 68% and +1 = 88%.

*Agreeableness* from 40% to 100%, this means that the zero point will be 70%, the star points will be approximately -1.682 = 45% and +1.682 = 95%. Then we can calculate the variance interval which will be: +/-15 and the -1 = 55% and +1 = 85%.

From the explanations above follows that we can now write in our program (see the Figure below) the input values for the experiment.

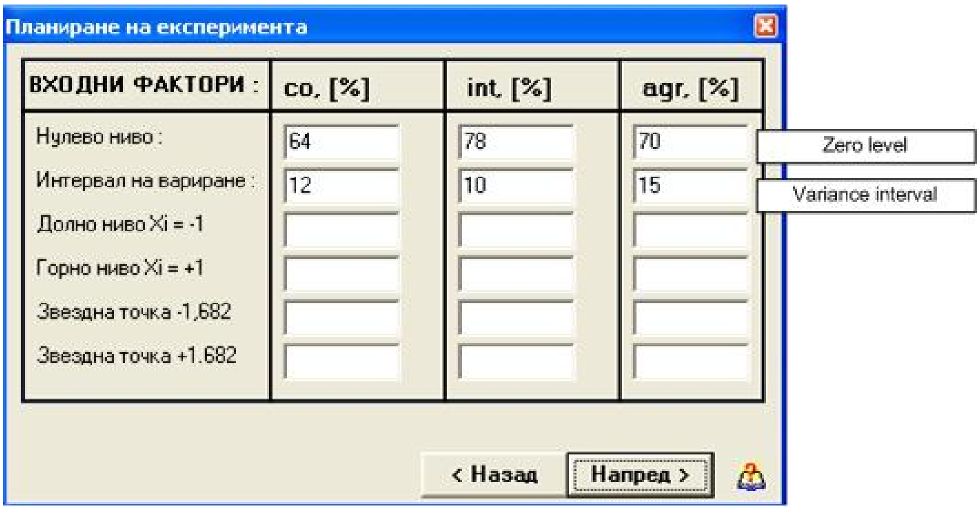



**Figure 39 Input values for the factors**

*I. The first experiment that we conduct is by motivation of 55%*

The next screen (Figure 40) shows us the entire table of the experiment (already explained in the previous part of this chapter). We have alltoghether 20 experiments (all of them displaied with their coded and natural values): 8 of them are in the 8 possible combinations of the three factors; 6 are in the star points and the rest 6 are in the zero-point. We see the factors $X_1$ , $X_2$ and $X_3$ first in their coded view and after this with their real experiment values and in the last column we add the value for the observed/resultant factor – the performance. We insert the measured performance values and on the next step we will see if they match with the calculated from the programm.

План на експеримента и резултати от експер

Plan of the experiment

Measured performance

| No | Кодирани фактори | | | План на експеримента | | | Данни от експеримента : |
|---|---|---|---|---|---|---|---|
| експер. | X1 | X2 | X3 | co, [%] | int, [%] | agr, [%] | pr, [%] |
| 1 | -1 | -1 | -1 | 52.000 | 68.000 | 55.000 | 44.29 |
| 2 | 1 | -1 | -1 | 76.000 | 68.000 | 55.000 | 51 |
| 3 | -1 | 1 | -1 | 52.000 | 88.000 | 55.000 | 52 |
| 4 | 1 | 1 | -1 | 76.000 | 88.000 | 55.000 | 56.9 |
| 5 | -1 | -1 | 1 | 52.000 | 68.000 | 85.000 | 51.44 |
| 6 | 1 | -1 | 1 | 76.000 | 68.000 | 85.000 | 58.83 |
| 7 | -1 | 1 | 1 | 52.000 | 88.000 | 85.000 | 58.25 |
| 8 | 1 | 1 | 1 | 76.000 | 88.000 | 85.000 | 64.88 |
| 9 | -1.682 | 0 | 0 | 43.816 | 78.000 | 70.000 | 55 |
| 10 | 1.682 | 0 | 0 | 84.184 | 78.000 | 70.000 | 66.15 |
| 11 | 0 | -1.682 | 0 | 64.000 | 61.180 | 70.000 | 51 |
| 12 | 0 | 1.682 | 0 | 64.000 | 94.820 | 70.000 | 61.35 |
| 13 | 0 | 0 | -1.682 | 64.000 | 78.000 | 44.770 | 43.29 |
| 14 | 0 | 0 | 1.682 | 64.000 | 78.000 | 95.230 | 55.64 |
| 15 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 69.5 |
| 16 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 69.9 |
| 17 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 69.6 |
| 18 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 69.2 |
| 19 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 69.8 |
| 20 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 69.1 |

< Назад

В Clipboard

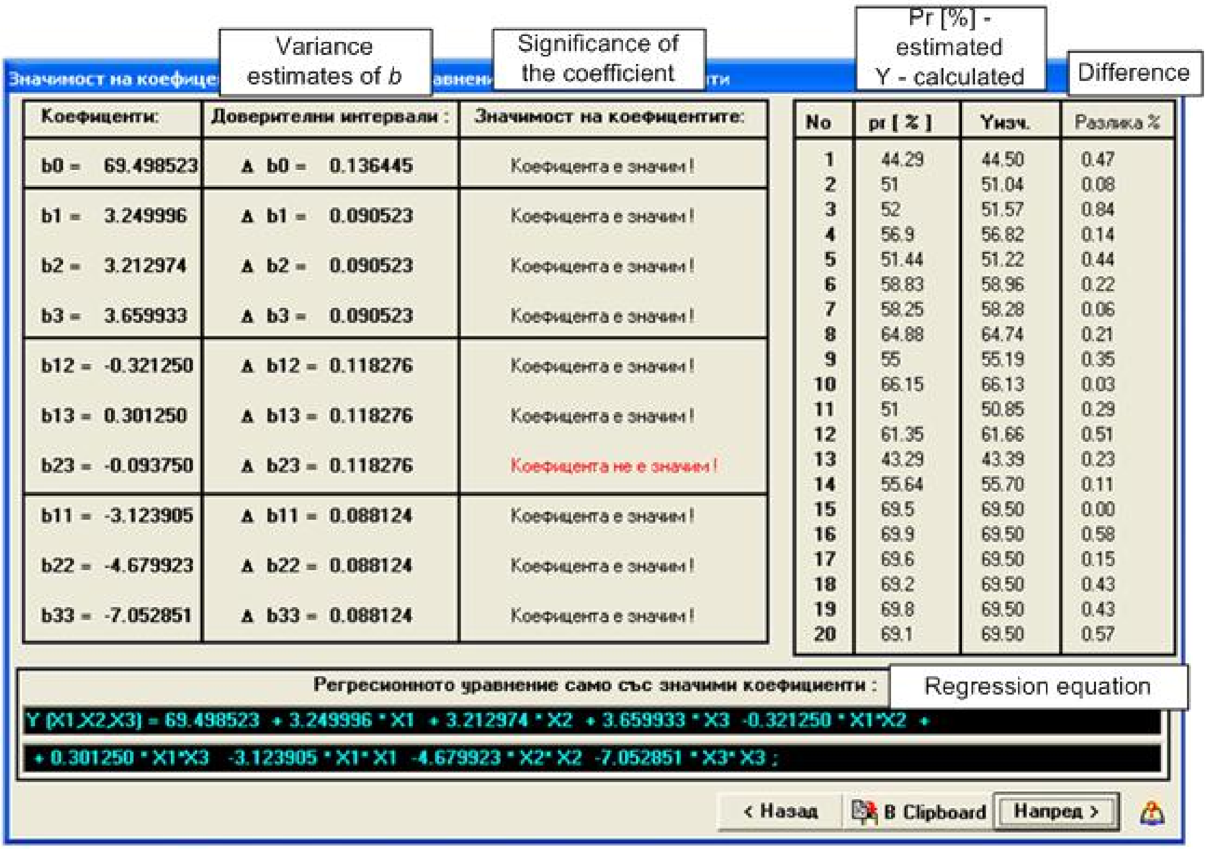
Запис ...

Напред >

**Figure 40 Plan and Results of the Experiment (Motivation=55%)**

The software is carrying out all the needed calculations and delivers the information visualized on the next screen (Figure 41). We first see all coefficients ***b*** in the regression equation, calculated according formulas (6.6) and after this we see $s^2[b]$ or Δ ***b*** - the estimates of the variances of the regression coefficients according to formulas (6.7). Having this data we can find out which of the coefficients are significant and which not. We see that we have only the $\boldsymbol{b_{23}}$ that is not statistically significant and because of this the X value connected with it will not be included in the end equation. The next part of the screen is occupied with the data for the output factor – the performance: we have our estimated values, after this follow the calculated values from the program and after this the difference between the two in percentage. This difference shows how near the calculated values compared to the experimental ones are. If the proposed data is very near to the experimental one this means that the possibility to obtain an adequate model is very high. And vice versa if the difference is high, then the possibility for adequate model is low. On the bottom of the screenshot, is shown the mathematical equation in coded form only with significant coefficients:

$$Y(X_1,X_2,X_3)= 69.498523 + 3.249996*X_1 + 3.212947*X_2 + 3.659933*X_3 - 0.321250*X_1*X_2 + 0.301250*X_1*X_3 - 3.123905*X_1*X_1 - 4.679923* X_2*X_2 - 7.052851* X_3*X_3 \quad (6.19)$$




| Коефиценти: | Доверителни интервали : | Значимост на коефициентите: |
|---|---|---|
| b0 = 69.498523 | Δ b0 = 0.136445 | Коефицента е значим ! |
| b1 = 3.249996 | Δ b1 = 0.090523 | Коефицента е значим ! |
| b2 = 3.212974 | Δ b2 = 0.090523 | Коефицента е значим ! |
| b3 = 3.659933 | Δ b3 = 0.090523 | Коефицента е значим ! |
| b12 = -0.321250 | Δ b12 = 0.118276 | Коефицента е значим ! |
| b13 = 0.301250 | Δ b13 = 0.118276 | Коефицента е значим ! |
| b23 = -0.093750 | Δ b23 = 0.118276 | Коефицента не е значим ! |
| b11 = -3.123905 | Δ b11 = 0.088124 | Коефицента е значим ! |
| b22 = -4.679923 | Δ b22 = 0.088124 | Коефицента е значим ! |
| b33 = -7.052851 | Δ b33 = 0.088124 | Коефицента е значим ! |

| No | pr [ % ] | Yизч. | Разлика % |
|---|---|---|---|
| 1 | 44.29 | 44.50 | 0.47 |
| 2 | 51 | 51.04 | 0.08 |
| 3 | 52 | 51.57 | 0.84 |
| 4 | 56.9 | 56.82 | 0.14 |
| 5 | 51.44 | 51.22 | 0.44 |
| 6 | 58.83 | 58.96 | 0.22 |
| 7 | 58.25 | 58.28 | 0.06 |
| 8 | 64.88 | 64.74 | 0.21 |
| 9 | 55 | 55.19 | 0.35 |
| 10 | 66.15 | 66.13 | 0.03 |
| 11 | 51 | 50.85 | 0.29 |
| 12 | 61.35 | 61.66 | 0.51 |
| 13 | 43.29 | 43.39 | 0.23 |
| 14 | 55.64 | 55.70 | 0.11 |
| 15 | 69.5 | 69.50 | 0.00 |
| 16 | 69.9 | 69.50 | 0.58 |
| 17 | 69.6 | 69.50 | 0.15 |
| 18 | 69.2 | 69.50 | 0.43 |
| 19 | 69.8 | 69.50 | 0.43 |
| 20 | 69.1 | 69.50 | 0.57 |

**Figure 41 Coefficients and the regression equation in coded form**

On the next screenshot – Figure 42 there is a lot of information, so let us start with the first rows: there we see the Variance of adequacy – according to formulas 6.8 and 6.10 and the Fisher Criterion $F = \frac{S^2_{adequacy}}{S^2_{reproducibility}}$. The Fisher Criterion is $F=0.98972 < F_{table}= 6.09$ (Dean & Voss 1999*)* and *this means that the resultant mathematical model is adequate and statistically correct.*

Subsequent comes the regression equation in natural form that describes the searched from us model. ***Having this equation we can predict the performance of every employee based only on his psychological features (motivation, conscientiousness, intelligence and agreeableness).***

The model looks like:

*Performance* [%] by Motivation of 55% = pr(co, int, agr) = -523.021607 + 3.139297*co + 7.793311*int + 4.525325*agr – 0.002677*co*int + 0.001674*co*agr – 0.021694*co*co – 0.046799*int*int – 0.031346*agr*agr (6.20)

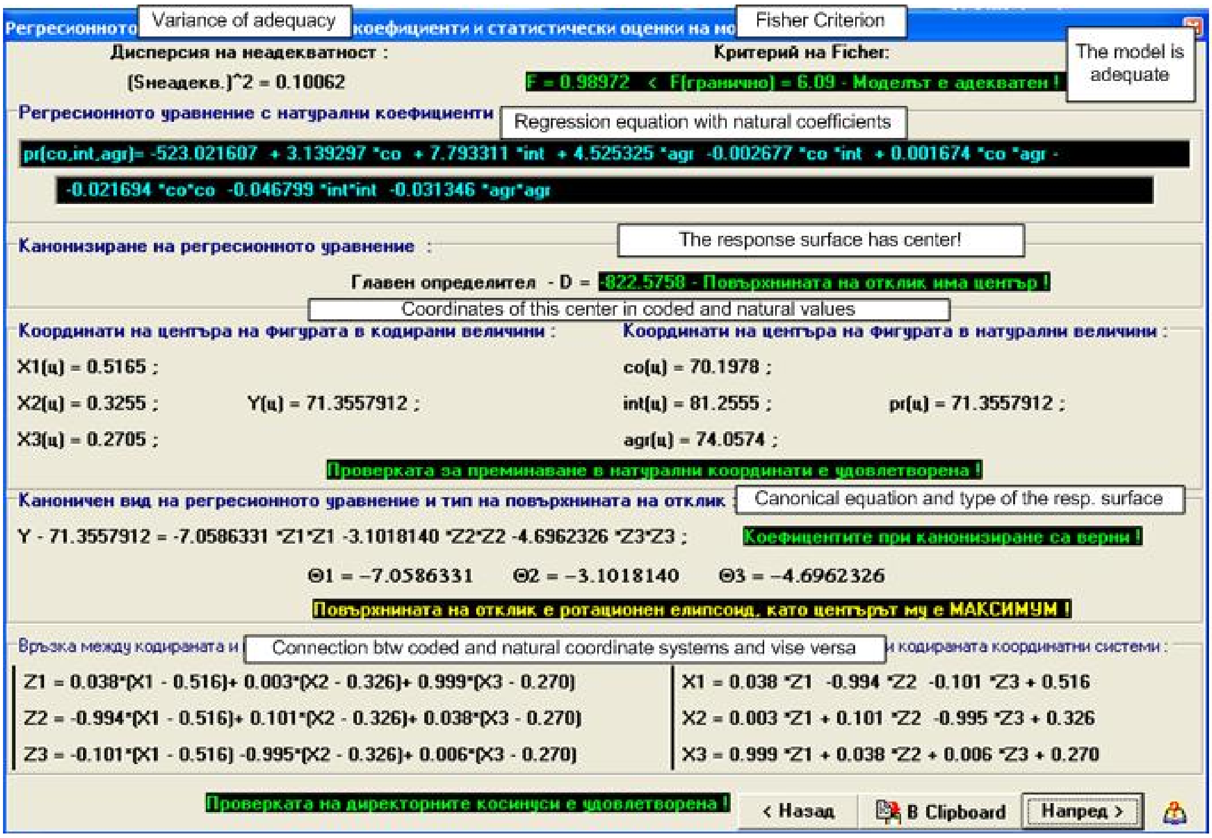



**Figure 42 Regression equation in natural form and statistical analysis of the model**

After having the mathematical model now we have to find the figure of the response surface and it central point. For this we are using D= -822.5758 (D is the matrix discriminant of the coefficients of the model) which gives us the information that the response surface has a center and after this the program gives us the calculated coded and natural values for this central point. Later on comes the canonical form of the regression equation and the check if it is correct and this canonization gives us the information that the response surface has the form of rotational ellipsoid and that the center is its maximum. At the end come the equations that give us the connection between coded and natural coordinate systems and vice versa and a check that proves that the canonical transformation is correct.

On Figure 43 we see the two-dimensional intersections for all the possibilities (-1, 0, 1) for each of the input factors ($X_1$, $X_2$, $X_3$). For every cut we have a 3D graphic in MATLAB on which we can see how exactly the response surface looks like and where exactly the maximum is.

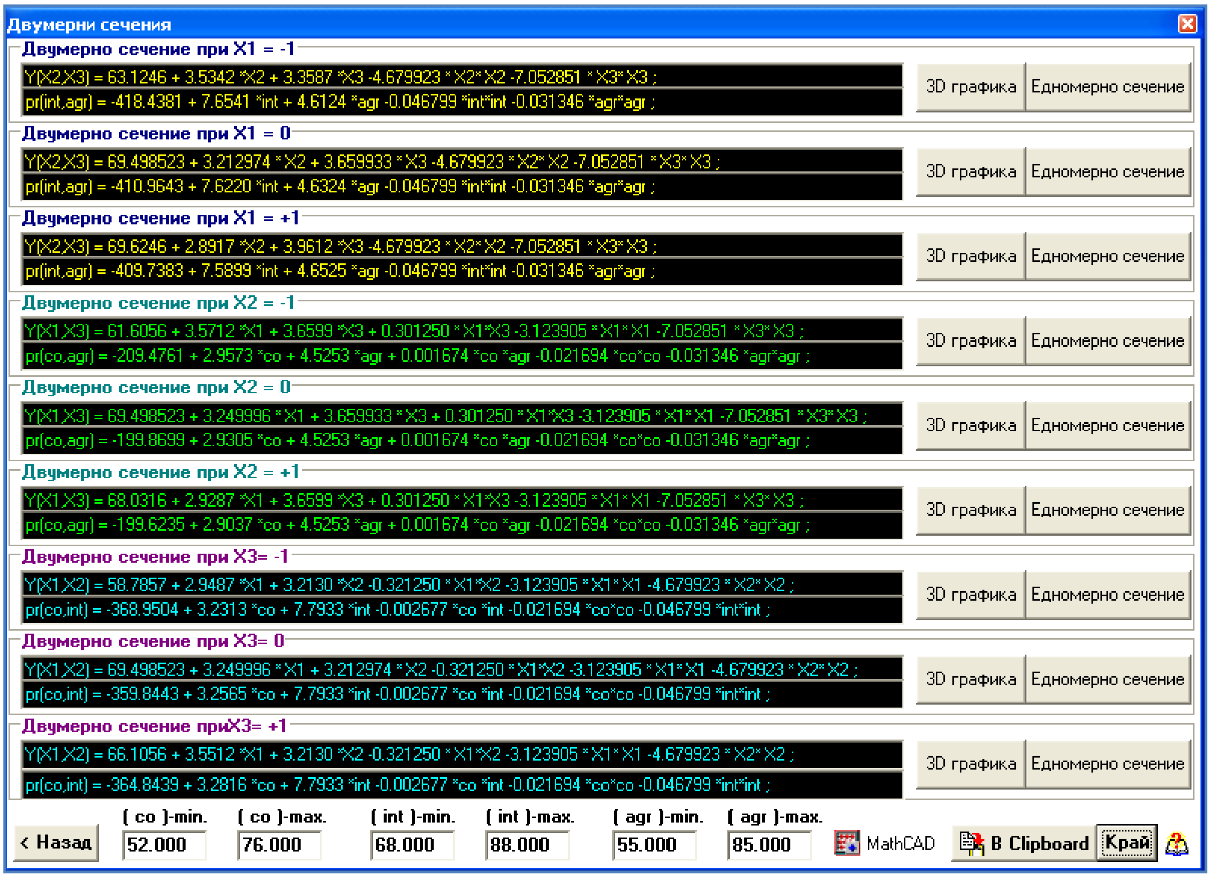



**Figure 43 Two-dimensional intersections**

For example on the following figures we see the intersection for agreeableness and openness/intelligence where the other two factors (*motivation*=55% and *conscientiousness*=52%) are fixed because otherwise we cannot display the graphic on a 3D figure. We will explain now the first three graphics for the 2-dimensional cut for agreeableness and intelligence, which we gain from the yellow equations from figure 42. $X_1$ is the conscientiousness and the values of -1, 0 and +1 are actually the natural values of 52%, 64% and 76%.

On Figure-44 a) we can see the response surface of the performance as function of agreeableness and intelligence, where agreeableness changes btw 40% and 100% and intelligence btw 60% and 100%. The maximum point of the surface is by approximately 75% of agreeableness and 80% of intelligence and is exactly 64%.

***By 60% of intelligence***, the connection btw agreeableness and performance is an ellipse. The minimum values of performance of 7% are by agreeableness of 40%, when agreeableness grows until 75% we gain the maximum values of performance of 42%, by agreeableness of 80% the performance is 40.7%. The further growing of agreeableness until 100% leads to decreasing of the performance values up to 20%. This can be explained with the specific influence of this psychometric characteristic over the personal performance: the growing values of agreeableness up to 75% characterize with growing of the performance because the employee is able to communicate and cooperate with his colleagues, he is able to accept others' ideas and to follow instructions; after these values the person loses his own judgment and cannot resolve any problem alone. The software engineer agrees with everyone and is not able to take decisions anymore and this leads to low values of performance. When the values are around 40% means that he is not able to cooperate and works very difficult with other people and this of course means also low productivity.

The next observations that we will explain are about the influence of the intelligence over the performance. On Figure-44 a) by fixed agreeableness of 40% we can see that by *intelligence* of 60% the *performance* is only 7% and with growing of the intelligence values up to 80% we reach performance of around 29%. The further growing of the intelligence leads to decreasing of the performance values up to 13.3%. The observations have shown that with the increasing of the intelligence after a specific point (around 80%), the observed software team members start to take very complex decisions and don't choose the optimal algorithm for resolving a problem. This leads to complications and more mistakes in the work process, the employees need more time and the solutions are not optimal, because of this it is logical to observe the decreasing of the performance (productiveness). By low values, even by 60% openness we have very low productiveness which shows that we need employees with intelligence over the average in order to manage the software engineering process.

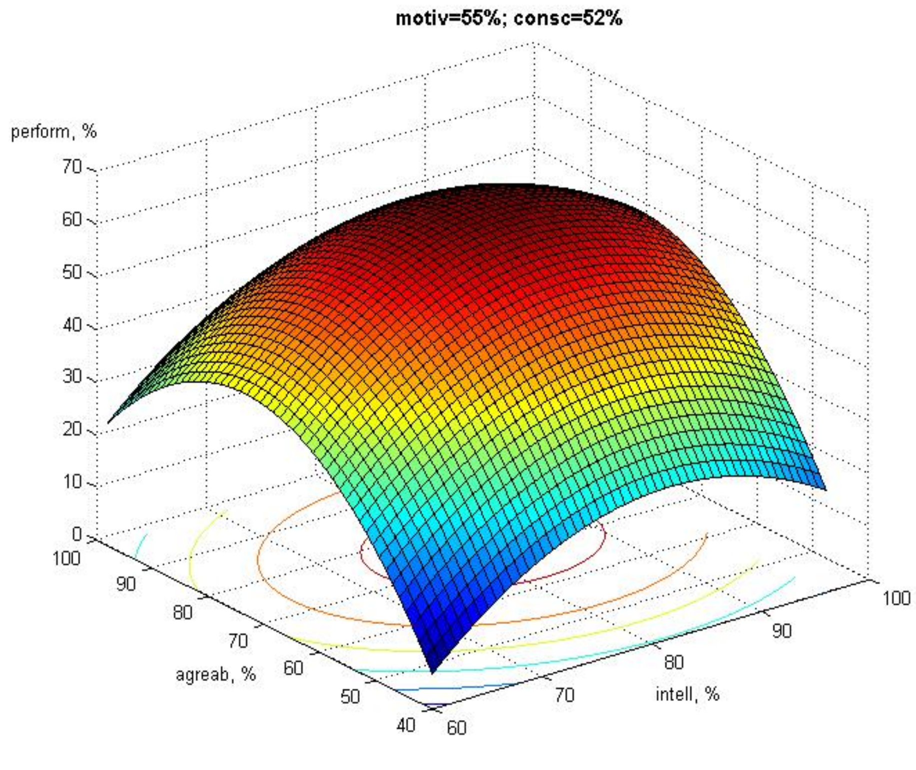



**Figure 44 a) Response surface by (motivation=55% and conscientiousness=52%)**

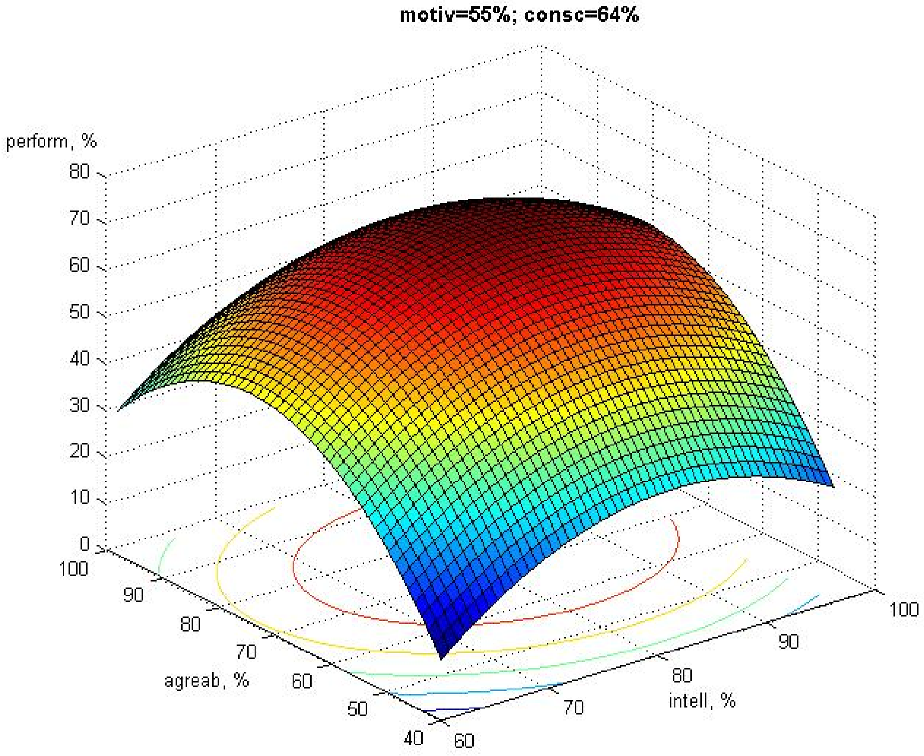



**Figure 44 b) Response surface by (motivation=55% and conscientiousness=64%)**

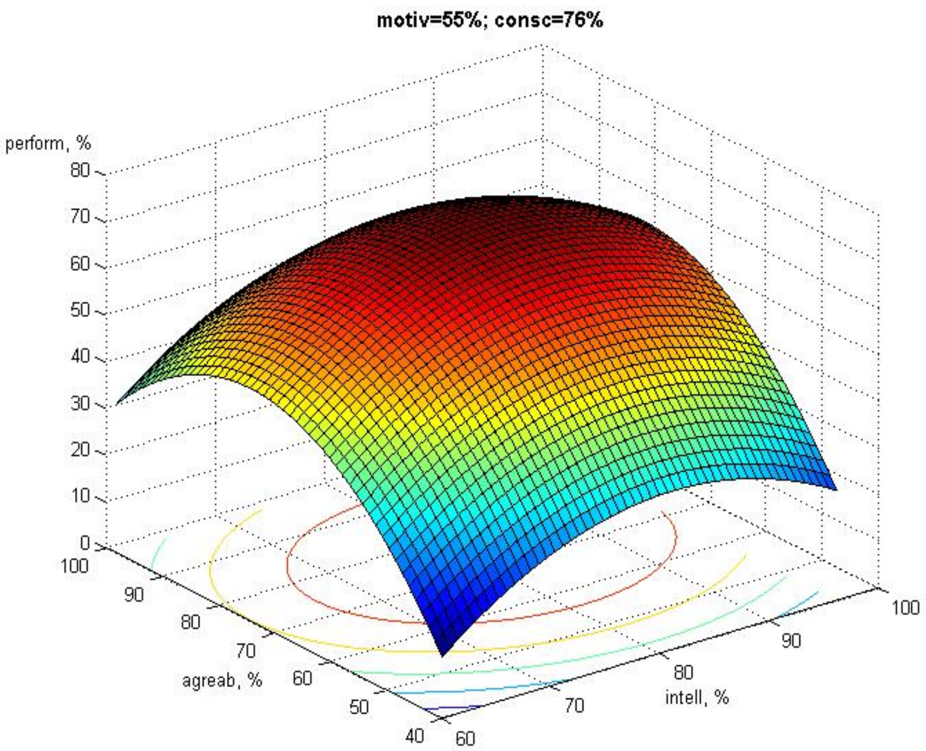



**Figure 44 c) Response surface by (motivation=55% and conscientiousness=76%)**

On the Figures 44 –b) and 44 –c) can be observed the same experimental dependencies, which can be analogically explained. Because of this we are going to give only the specific points from the response surface.

For example in 44 –c) which is by conscientiousness of 76% we have that by intelligence of 60% and agreeableness of 40% the performance is only 13%. The maximum of performance is by agreeableness of 80% and is 49%. The further growing of agreeableness until 100% leads to reduction in the productiveness value to 29%. When observing the intelligence values we can see that by 60% the performance is only 13% and by increasing the values up to 83% we gain the maximum point of around 34% productiveness. The next increasing of the intelligence up to 100% results with 17% performance.

Moving in intersection by the values of conscientiousness of 76%, agreeableness of 75% and of intelligence 83% we can find the maximum performance value showed on Figure 44 –c) and it is 70%.

Having these explanations we have observed the influence of ***intelligence and agreeableness*** over the performance, when the values for conscientiousness and motivation are fixed. Actually we have observed the one-dimensional intersections of the corresponding dependencies for better understanding of the changing values.

In order to describe also the influence of the conscientiousness over the productiveness we are using the next figures (origin are the second green group of formulas on figure 43) where the intelligence takes values of 68%, 78% and 88% and the motivation is fixed on 55%.

On the next figures 45 a) to 45 c) is shown the influence of ***agreeableness and conscientiousness*** over the performance, where agreeableness changes btw 40% and 100% and conscientiousness btw 40% and 90%. The maximum point of the surface is by approximately 75% of agreeableness and 70% of conscientiousness and is exactly 63%. The motivation is fixed to 55% and the intelligence takes three particular values.

We are explaining only the connection btw ***conscientiousness and performance*** and as we see it is again an ellipse with the following important points (Fig. 45a): by agreeableness of 40% we have values for conscientiousness btw 40% and 90%. By conscientiousness btw 40% to 55% we have very low performance of about 8% to 22%. When the conscientiousness values are growing up to 70% we have the maximum values of performance of 27% and after this by conscientiousness of 80% we have 24.4% performance and by conscientiousness of 90% we have 18% performance. This can be explained with the specific of this psychological characteristic and it is that by growing until 70% it means that the software specialist is trying to do his best and to manage his work as good as possible. From other side this characteristic hinders the process of ignoring the unimportant details in the everyday work, and exactly this leads to decreasing of the performance, when the conscientiousness is higher than 70%. The employee loses too much time in checking details and spending time for not so important problems which needs more time and results into lower productivity. When the values are low, until 55% we have very low performance and this is to be explained with the fact that such employees are not doing their job with the needed respect and cautious.

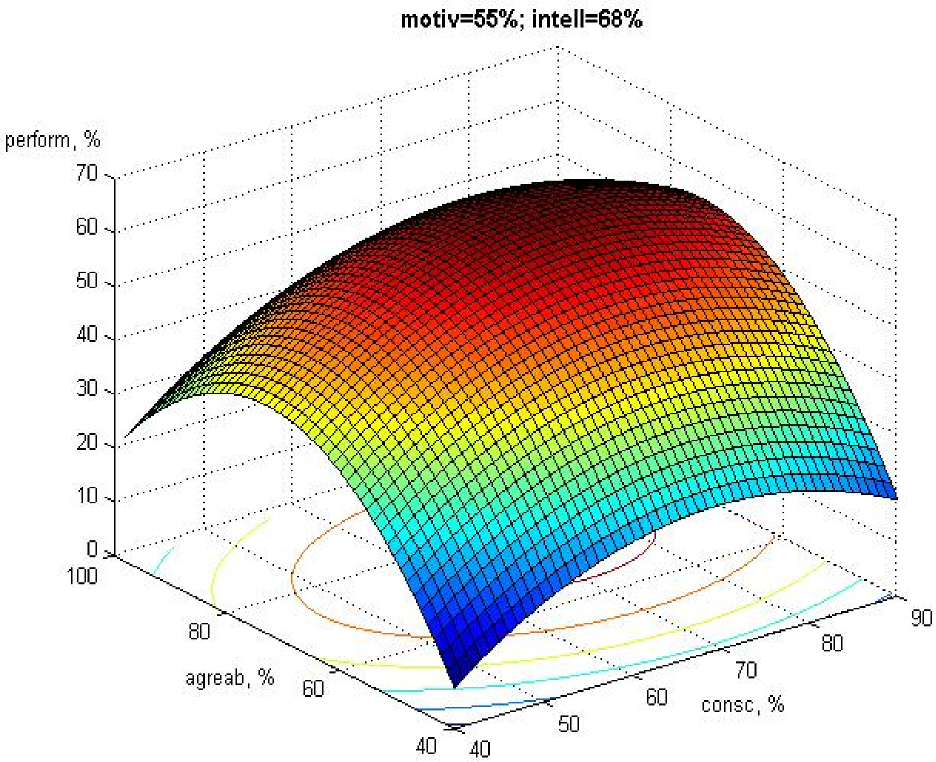



**Figure 45 a) Response surface by (motivation=55% and intelligence=68%)**

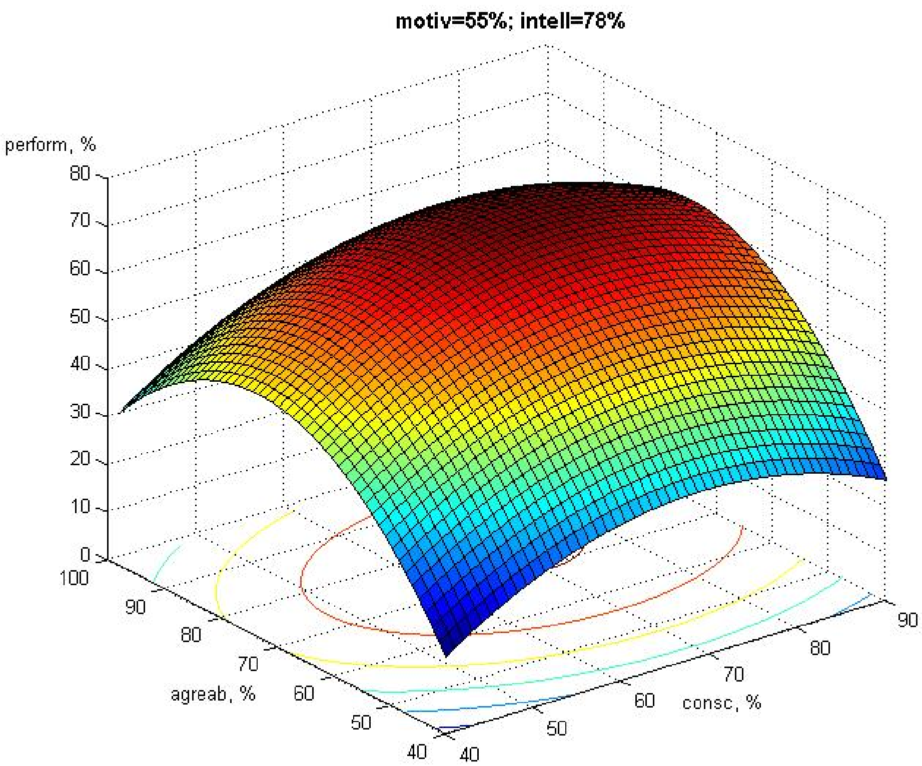



**Figure 45 b) Response surface by (motivation=55% and intelligence=78%)**

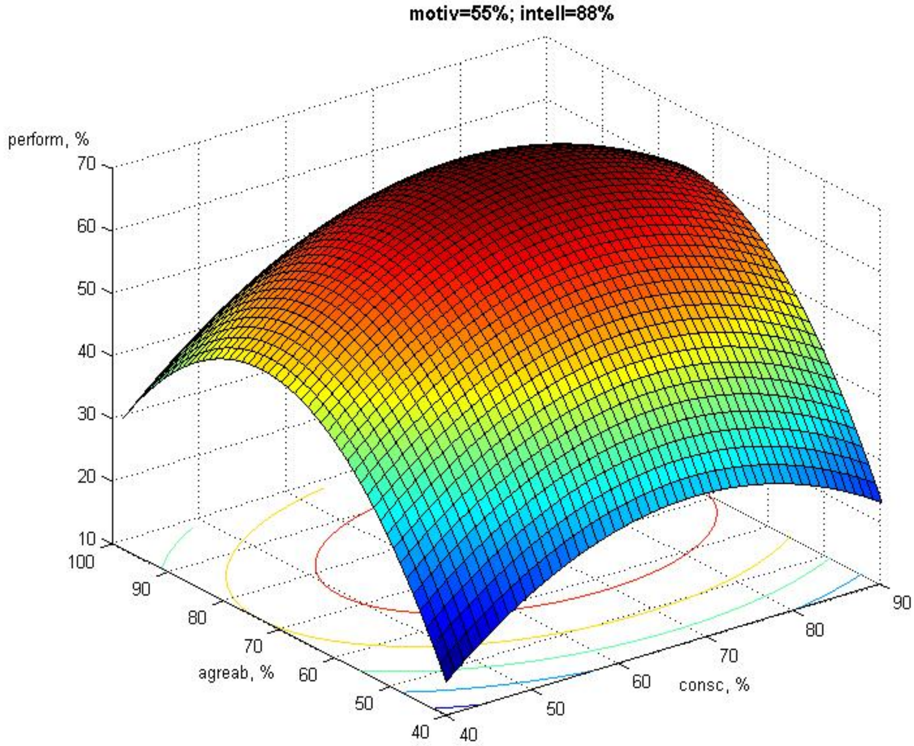



**Figure 45 c) Response surface by (motivation=55% and intelligence=88%)**

On the Figures 45 –b) and 45 –c) can be observed the same experimental dependences and because of this we will give only some values. For example in 45 –c) which is by intelligence of 88% we have that by conscientiousness of 40% and agreeableness of 40% the performance is only 15.3%. The maximum of performance is by conscientiousness of 68% and is 33%. The further growing of conscientiousness until 90% leads to reduction in the productiveness value to 29%.

Moving in intersection by the values of intelligence of 88%, conscientiousness 68% and of agreeableness 75% we can find the maximum performance value showed on Figure 45–c) and it is 69%.

The rest three figures (46 a, b, c), originating from the turquoise equations (Figure 43) are absolutely analogical to the previous ones, the only difference is that here is visualized the dependence btw intelligence, conscientiousness and performance where the other factors motivation=55% and agreeableness=55%, 70%, 85% are fixed. We will not explain the dependencies once again as we have already said they are the same as in the other figures. We are giving the figures just for better understanding.

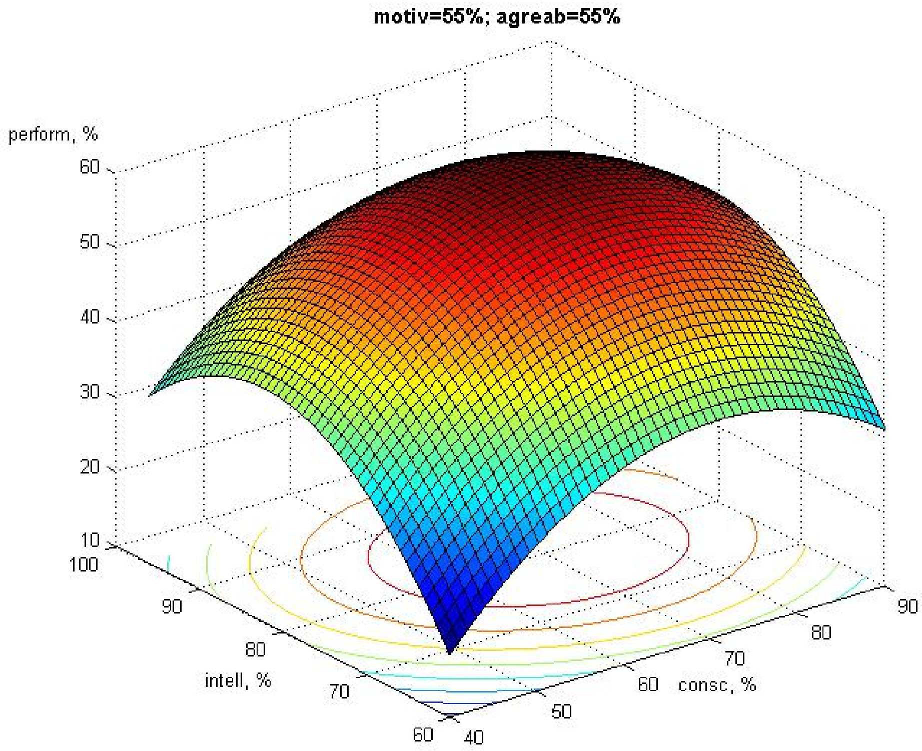



**Figure 46 a) Response surface by (motivation=55% and agreeableness=55%)**

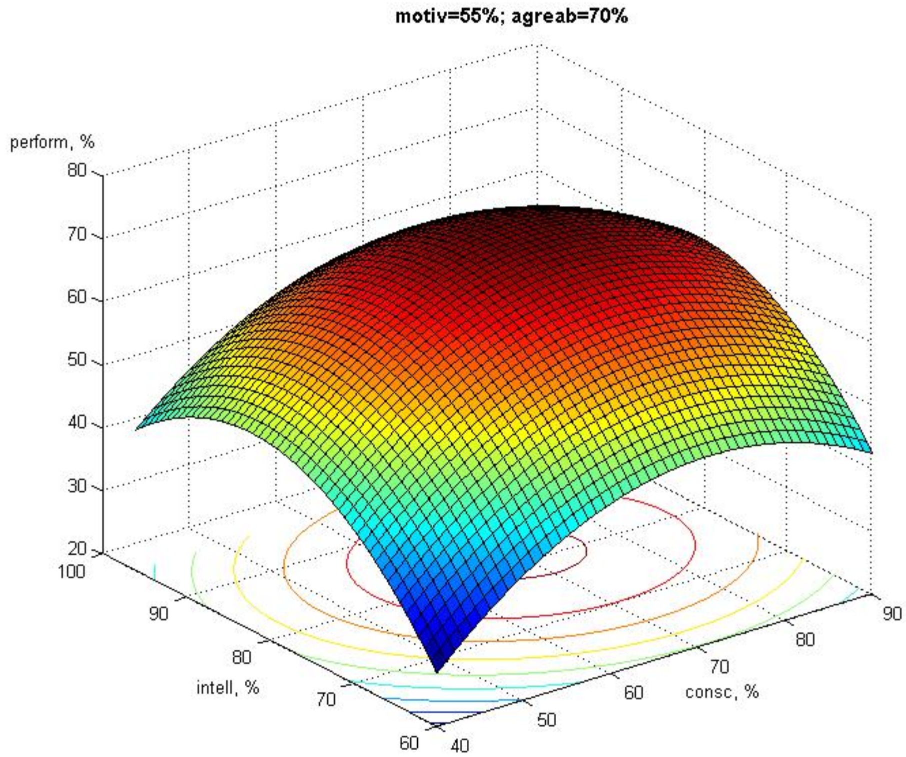



**Figure 46 b) Response surface by (motivation=55% and agreeableness=70%)**

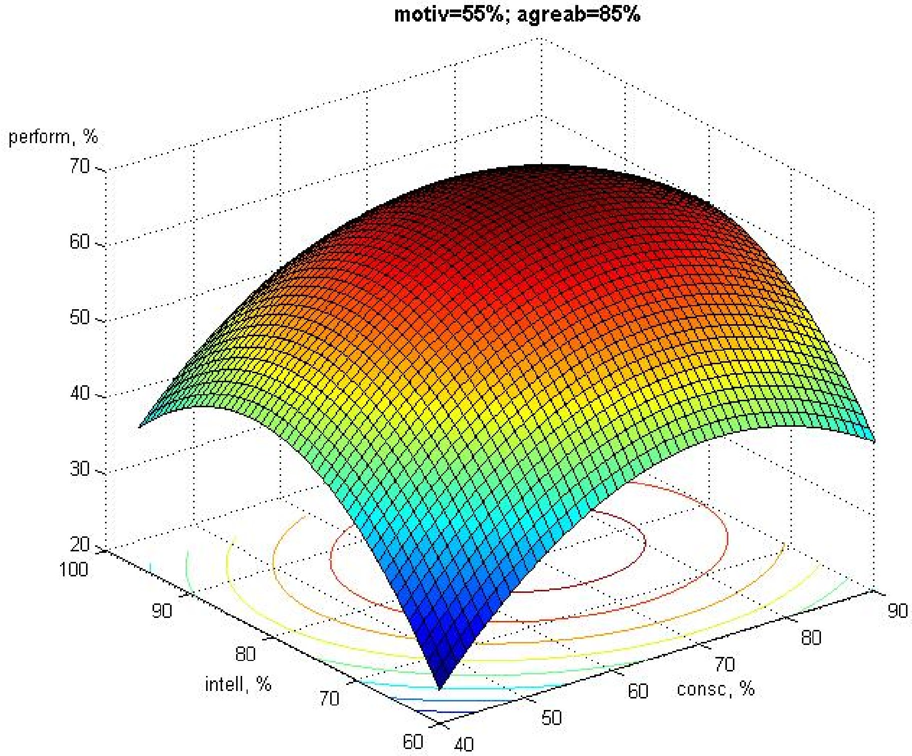



**Figure 46 c) Response surface by (motivation=55% and agreeableness=85%)**

*II. The second experiment is by motivation of 70%.*

We will not explain once again the first three screen-shots from the program that are about the input data because they are every time the same. We will continue with Figure 47, where we can see the plan of the experiment with the 20 experiments (as already explained 8 of them are in the 8 possible combinations of the three factors; 6 are in the star points and the rest 6 are in the zero-point). We see the factors $X_1$ , $X_2$ and $X_3$ with their coded and with their real values and in the last column we add the value for the observed/resultant factor – the performance, but this time by motivation of 70%.

План на експеримента и резултати от експе | Plan of the experiment | Measured performance

| No експер. | Кодирани фактори X1 | X2 | X3 | План на експеримента co. [%] | int. [%] | agr. [%] | Данни от експеримента : pr. [%] |
|---|---|---|---|---|---|---|---|
| 1 | -1 | -1 | -1 | 52.000 | 68.000 | 55.000 | 52.7 |
| 2 | 1 | -1 | -1 | 76.000 | 68.000 | 55.000 | 60.6 |
| 3 | -1 | 1 | -1 | 52.000 | 88.000 | 55.000 | 60.8 |
| 4 | 1 | 1 | -1 | 76.000 | 88.000 | 55.000 | 67.7 |
| 5 | -1 | -1 | 1 | 52.000 | 68.000 | 85.000 | 61.2 |
| 6 | 1 | -1 | 1 | 76.000 | 68.000 | 85.000 | 70 |
| 7 | -1 | 1 | 1 | 52.000 | 88.000 | 85.000 | 69.3 |
| 8 | 1 | 1 | 1 | 76.000 | 88.000 | 85.000 | 77.2 |
| 9 | -1.682 | 0 | 0 | 43.816 | 78.000 | 70.000 | 65.5 |
| 10 | 1.682 | 0 | 0 | 84.184 | 78.000 | 70.000 | 78.7 |
| 11 | 0 | -1.682 | 0 | 64.000 | 61.180 | 70.000 | 60.7 |
| 12 | 0 | 1.682 | 0 | 64.000 | 94.820 | 70.000 | 73 |
| 13 | 0 | 0 | -1.682 | 64.000 | 78.000 | 44.770 | 51.5 |
| 14 | 0 | 0 | 1.682 | 64.000 | 78.000 | 95.230 | 66.2 |
| 15 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 82.7 |
| 16 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 82 |
| 17 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 82.8 |
| 18 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 82.4 |
| 19 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 82.3 |
| 20 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 82.2 |

< Назад | В Clipboard | Запис ... | Напред >

**Figure 47 Plan and Results of the Experiment (Motivation=70%)**

If we compare the performance values we will see a significant difference, here we have performance values of 82,8% and in the previous experiment we had 69,9%. So we can make the first observation that by enhancing the motivation the performance also grows significantly.

On the next figure 48 we see all coefficients ***b*** in the regression equation and the Δ ***b*** - the estimates of the variances of the regression coefficients. Having this data we can observe which of the coefficients are significant and which not. We see that again only one coefficient $b_{23}$ is not significant. The next part of the screen is occupied with the data for the output factor – the performance: we have our estimated values, after this follow the calculated values from the program and after this the difference between them in percentage. We can see that again as in the first experiment the difference is very small and we can end with the following regression equation in coded form with significant coefficients:

$$Y(X_1,X_2,X_3)= 82.384068 + 3.932305*X_1 + 3.748234*X_2 + 4.439234*X_3 - 0.237500*X_1*X_2 + 0.237500*X_1*X_3 - 3.633999*X_1*X_1 - 5.490612* X_2*X_2 - 8.319736* X_3*X_3 \qquad (6.21)$$

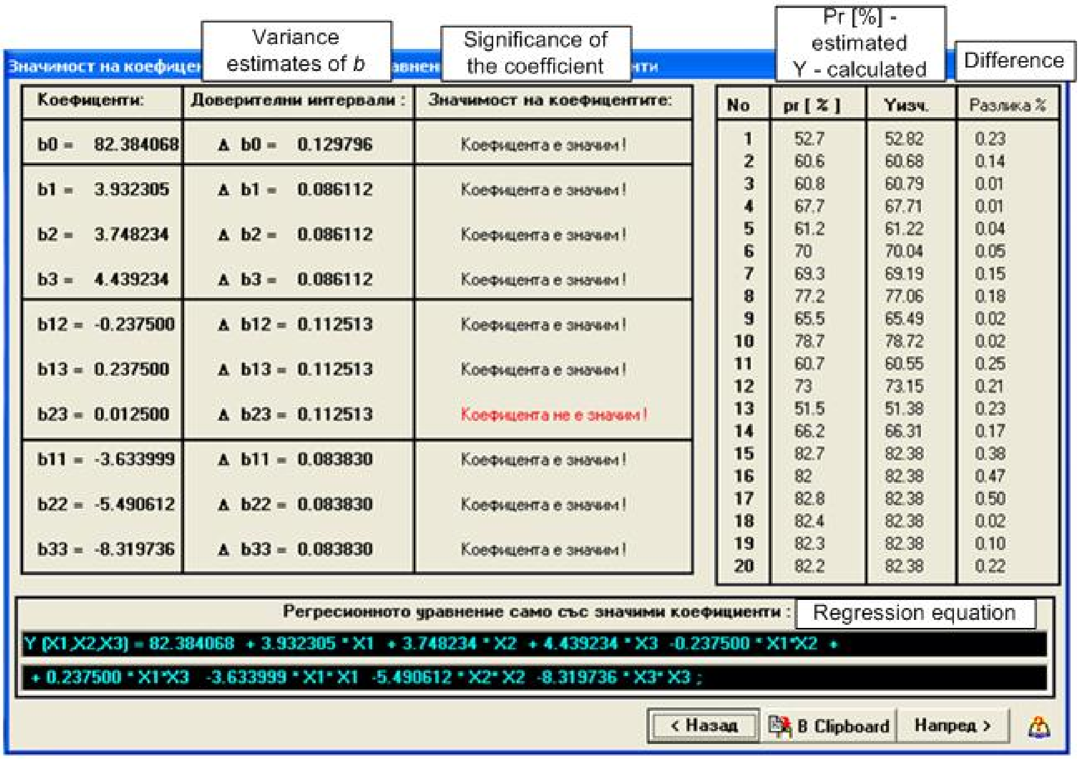



**Figure 48 Coefficients and the regression equation in coded form**

On the next screenshot – Figure 49 we see the Variance of adequacy, then the Fisher Criterion.

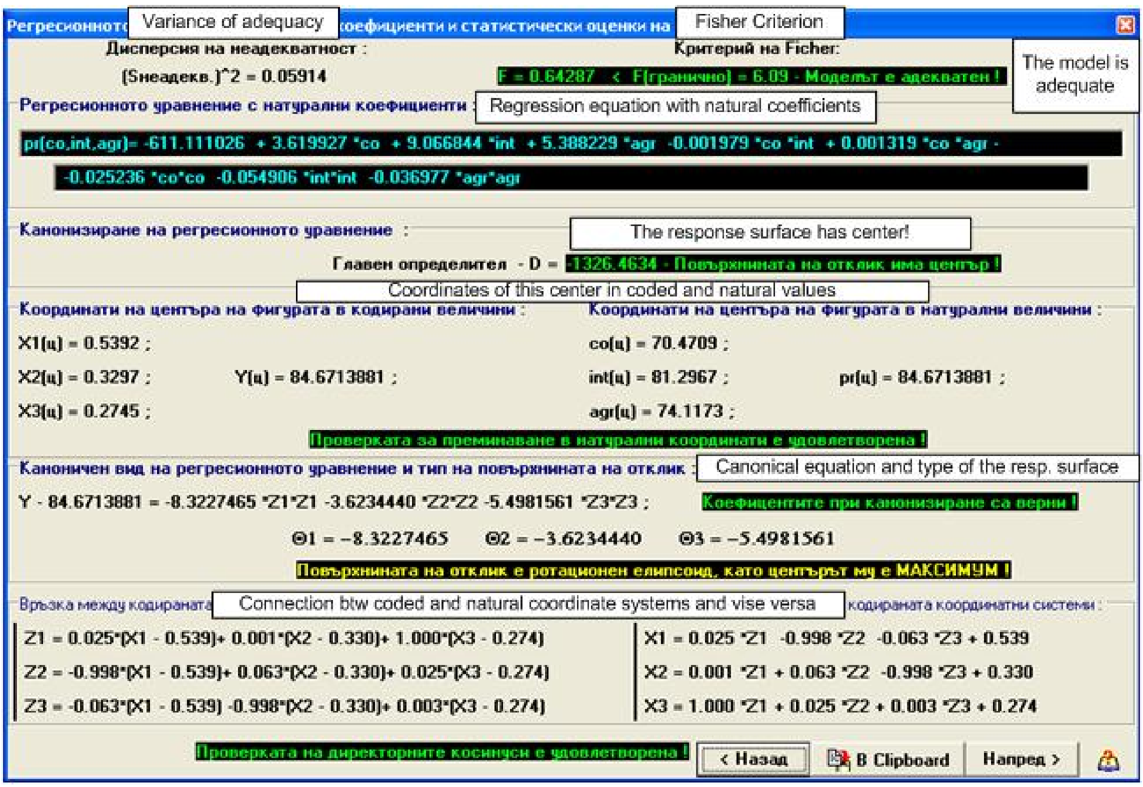



**Figure 49 Regression equation in natural form and statistical analysis of the model**

This statistical analysis shows that the gained model is correct and adequate and we have the regression equation in natural form that describes the ***performance of employees based only on their psychological features.***

*Performance* [%] by Motivation of 70% = pr(co, int, agr) = -611.111026 + 3.619927*co + 9.066844*int + 5.388229*agr – 0.001979*co*int + 0.001319*co*agr – 0.025236*co*co – 0.054906*int*int – 0.036977*agr*agr

(6.22)

On the next part of the screenshot follows the analysis for the response surface: if it has center, and if so, then what are the coordinates. They are displayed in coded and natural form. Later on are shown the canonical equations and the proof that the translation between the different coordinate systems is correct. This canonization gives us the information that the response surface has the form of rotational ellipsoid and that the center point is its maximum.

On Figure 50 we see the two-dimensional intersections for all the possibilities (-1, 0, 1) for each of the input factors ($X_1$, $X_2$, $X_3$). For every cut we have a 3D graphic in MATLAB on which we can see how exactly the response surface looks like and where exactly the maximum is. As we have explained in detail these 9 graphics for the previous experiment, here we will show only a table (table 31) with the corresponding values for the current case. Any further explanations would be just repetition of everything said before. For the interested reader we have shown all the graphics by Motivation of 70% and 85% and also the one-dimensional intersections in the Appendix.

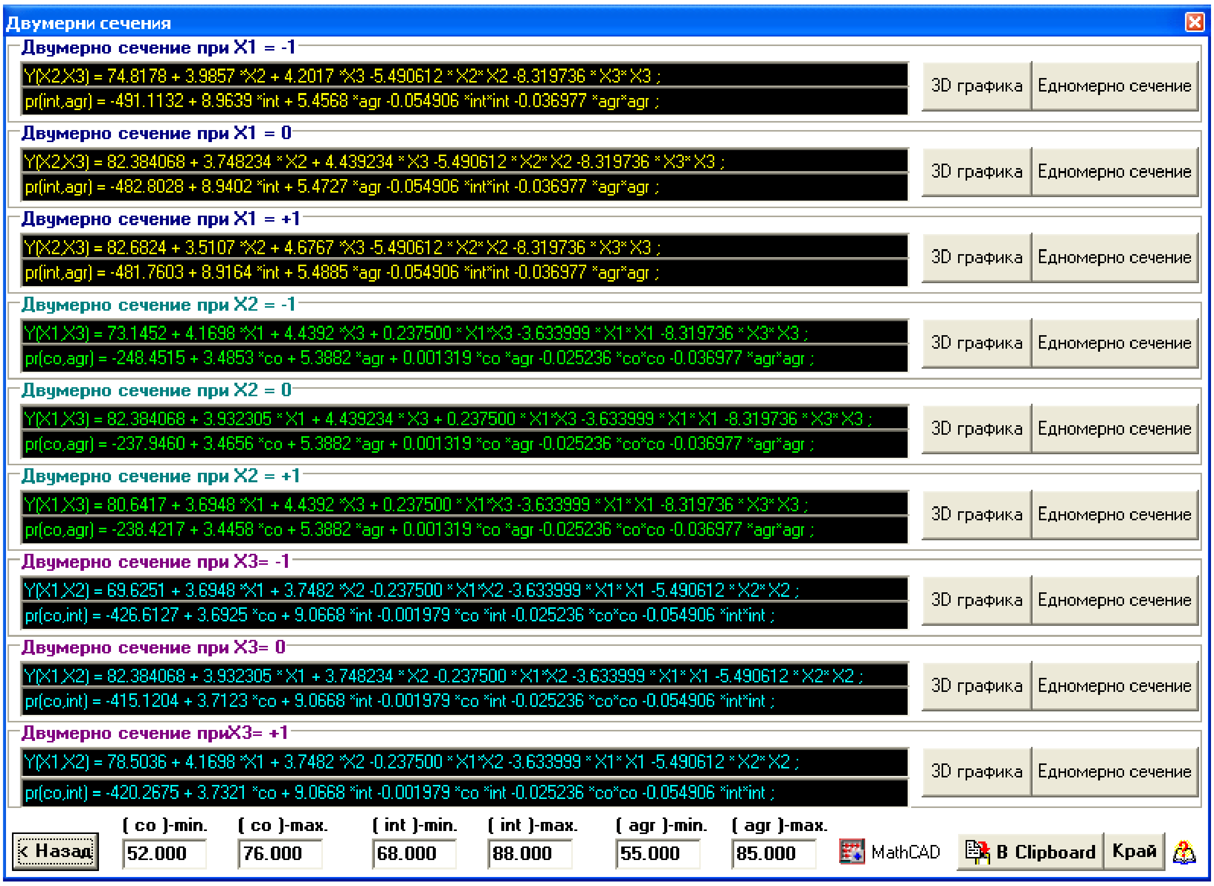



**Figure 50 Two-dimensional intersections**

**Table 31. Comparison between the performance values by motivation of 55% and of 70%, measured on the out lines of the factor space**

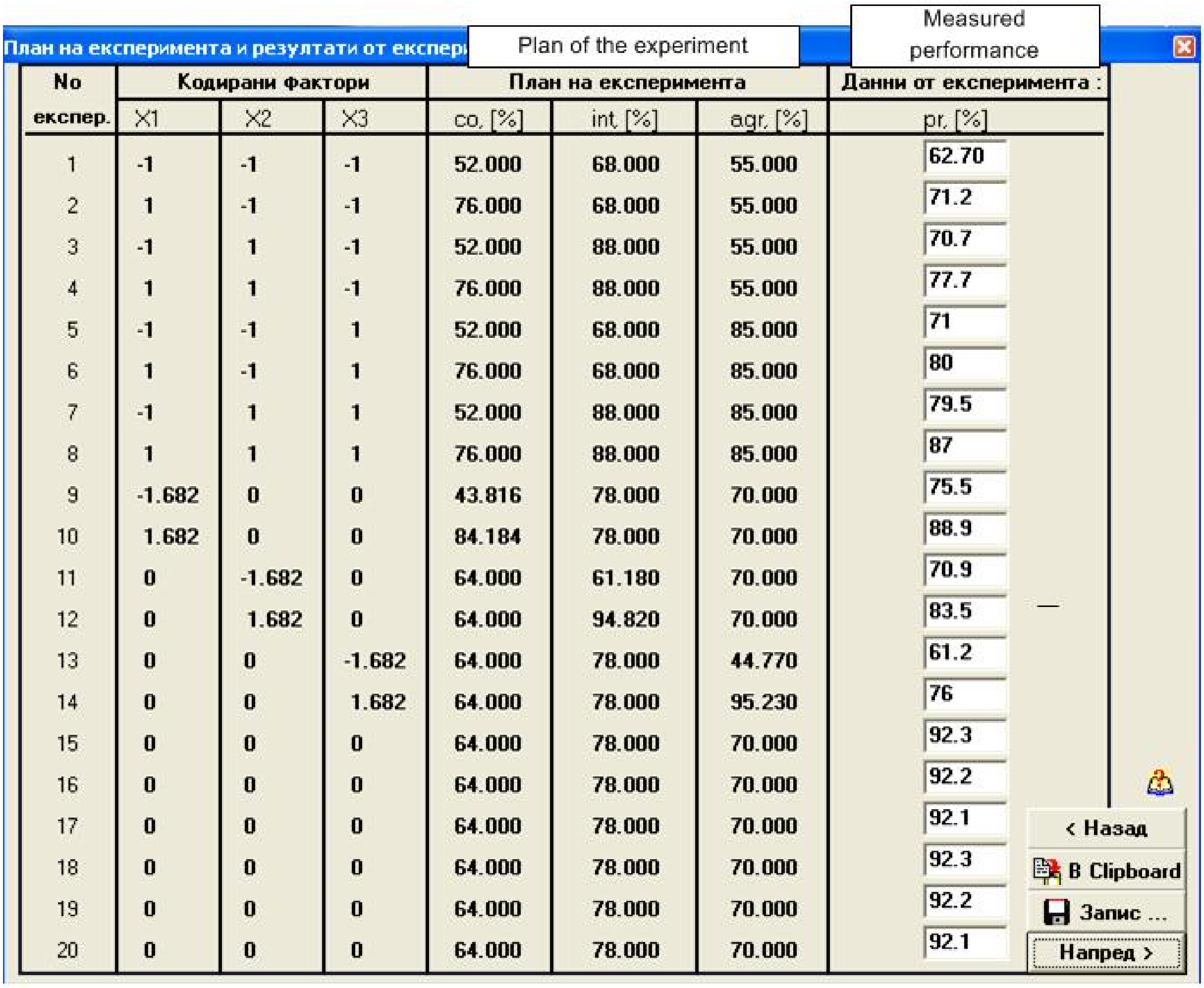
| Conscientiousness [%] | Intelligence [%] | Agreeableness [%] | By Motivation 55% Performance | By Motivation 70% Performance |
|---|---|---|---|---|
| 52 | 60 | 40 | 7 | 8,2 |
| | | 75 | 42 | 50,3 |
| | | 80 | 40,7 | 49 |
| | | 100 | 20 | 25 |
| | | | | |
| 52 | 60 | 40 | 7 | 8,2 |
| | 80 | | 29 | 33,7 |
| | 100 | | 13,3 | 15,3 |
| | | | | |
| 76 | 60 | 40 | 13 | 16 |
| | | 80 | 49 | 58 |
| | | 100 | 29 | 34,6 |
| | | | | |
| 76 | 60 | 40 | 13 | 16 |
| | 83 | | 34 | 40,4 |
| | 100 | | 17 | 21,2 |
| | | | | |
| 40 | 68 | 40 | 8 | 9 |
| 55 | | | 22 | 26 |
| 70 | | | 27 | 32 |
| 80 | | | 24,4 | 29,4 |
| 90 | | | 18 | 22 |
| | | | | |
| 40 | 88 | 40 | 15,3 | 17,5 |
| 68 | | | 33 | 39 |
| 90 | | | 23 | 28,4 |

The comparison between the performance values by motivation of 55% and of 70%, made on the out limits of the factor space takes values that can be seen from the figures 44, 45 and 46 and also from the rest figures in the Appendix.

We can clearly see in the table that we have significant increase in the performance values by motivation of 70%, but the dependencies of the different characteristics and the productivity stay the same as already explained for the previous experiment.

*III. The third experiment is by motivation of 85%*

We will start here directly with the explanation of the plan of the experiment shown on the figure 51 below as the other steps are the same as for the other two experiments. On the figure 51, we can see the plan of the experiment with the 20 experiments (as already explained 8 of them are in the 8 possible combinations of the three factors; 6 are in the star points and the rest 6 are in the zero-point). We see the factors $X_1$ , $X_2$ and $X_3$ with their coded and with their real values and in the last column we add the value for the observed/resultant factor – the performance, but this time by motivation of 85%.




| No експер. | Кодирани фактори X1 | X2 | X3 | План на експеримента co, [%] | int, [%] | agr, [%] | Данни от експеримента : pr, [%] |
|---|---|---|---|---|---|---|---|
| 1 | -1 | -1 | -1 | 52.000 | 68.000 | 55.000 | 62.70 |
| 2 | 1 | -1 | -1 | 76.000 | 68.000 | 55.000 | 71.2 |
| 3 | -1 | 1 | -1 | 52.000 | 88.000 | 55.000 | 70.7 |
| 4 | 1 | 1 | -1 | 76.000 | 88.000 | 55.000 | 77.7 |
| 5 | -1 | -1 | 1 | 52.000 | 68.000 | 85.000 | 71 |
| 6 | 1 | -1 | 1 | 76.000 | 68.000 | 85.000 | 80 |
| 7 | -1 | 1 | 1 | 52.000 | 88.000 | 85.000 | 79.5 |
| 8 | 1 | 1 | 1 | 76.000 | 88.000 | 85.000 | 87 |
| 9 | -1.682 | 0 | 0 | 43.816 | 78.000 | 70.000 | 75.5 |
| 10 | 1.682 | 0 | 0 | 84.184 | 78.000 | 70.000 | 88.9 |
| 11 | 0 | -1.682 | 0 | 64.000 | 61.180 | 70.000 | 70.9 |
| 12 | 0 | 1.682 | 0 | 64.000 | 94.820 | 70.000 | 83.5 |
| 13 | 0 | 0 | -1.682 | 64.000 | 78.000 | 44.770 | 61.2 |
| 14 | 0 | 0 | 1.682 | 64.000 | 78.000 | 95.230 | 76 |
| 15 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 92.3 |
| 16 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 92.2 |
| 17 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 92.1 |
| 18 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 92.3 |
| 19 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 92.2 |
| 20 | 0 | 0 | 0 | 64.000 | 78.000 | 70.000 | 92.1 |

**Figure 51 Plan and Results of the Experiment (Motivation=85%)**

If we compare the performance values with the other two experiments we will see the difference: in the first case we had performance of 69,9%; in the second 82,8% and here we have values of 92.3%. *This confirms our previous observation that by enhancing the motivation the performance also grows significantly.*

On the next figure 52 we see the coefficients ***b*** of the regression equation and their estimation of variance - Δ ***b.*** Having this data we can observe which of the coefficients are significant and which not. We see that here all coefficient are significant. The next part of the screen is occupied with the data for the performance: we have the estimated values, the calculated ones and after this the difference between them. We can see that again as in the other two experiments the difference is very small and we can end with the following regression equation in coded form with significant coefficients:

$$Y(X_1,X_2,X_3) = 92.183152 + 3.993549*X_1 + 3.748571*X_2 + 4.400294*X_3 - 0.375000*X_1*X_2 + 0.125000*X_1*X_3 + 0.125000*X_2*X_3 - 3.536538*X_1*X_1 - 5.304741*X_2*X_2 - 8.346049*X_3*X_3 \quad (6.23)$$

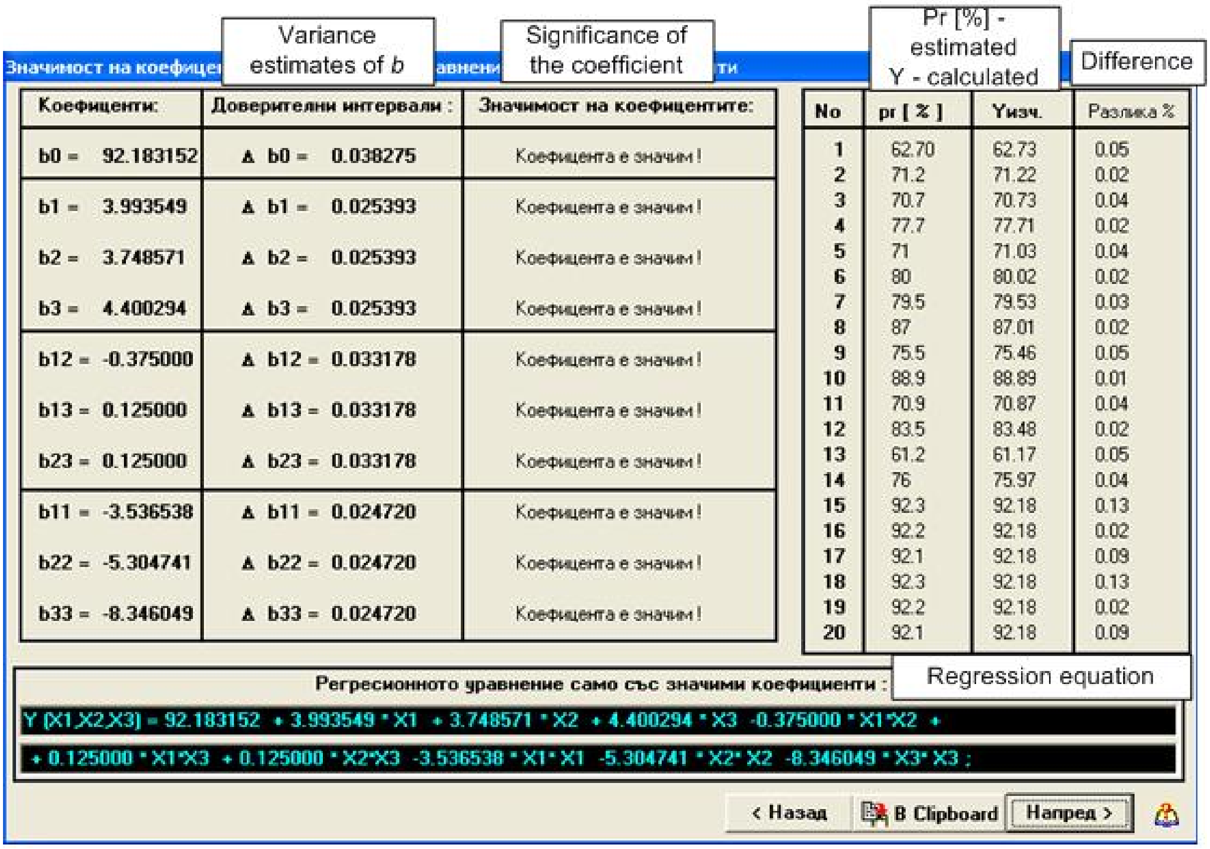



**Figure 52 Coefficients and the regression equation in coded form**

On the next screenshot – Figure 53 we see the Variance of adequacy and the Fisher Criterion. This statistical analysis shows that the gained model is correct and adequate and we have the regression equation in natural form that describes *the employees' performance by motivation of 85%.*

> *Performance* [%] by Motivation of 85% = pr(co, int, agr) = -591.921937 + 3.671524*co + 8.791920*int + 5.377006*agr – 0.003125*co*int + 0.000694*co*agr+0.000833*int – 0.024559*co*co – 0.053047*int*int – 0.037094*agr*agr
>
> (6.24)

On the next part of the screenshot follows the analysis for the response surface: if it has center, and if so, then what are the coordinates. They are displayed in coded and natural form. Later on are shown the canonical equations and the proof that the translation between the different coordinate systems is correct. This canonization gives us the information that the response surface has the form of rotational ellipsoid and that the center point is its maximum. Exactly like in the previous two experiments.

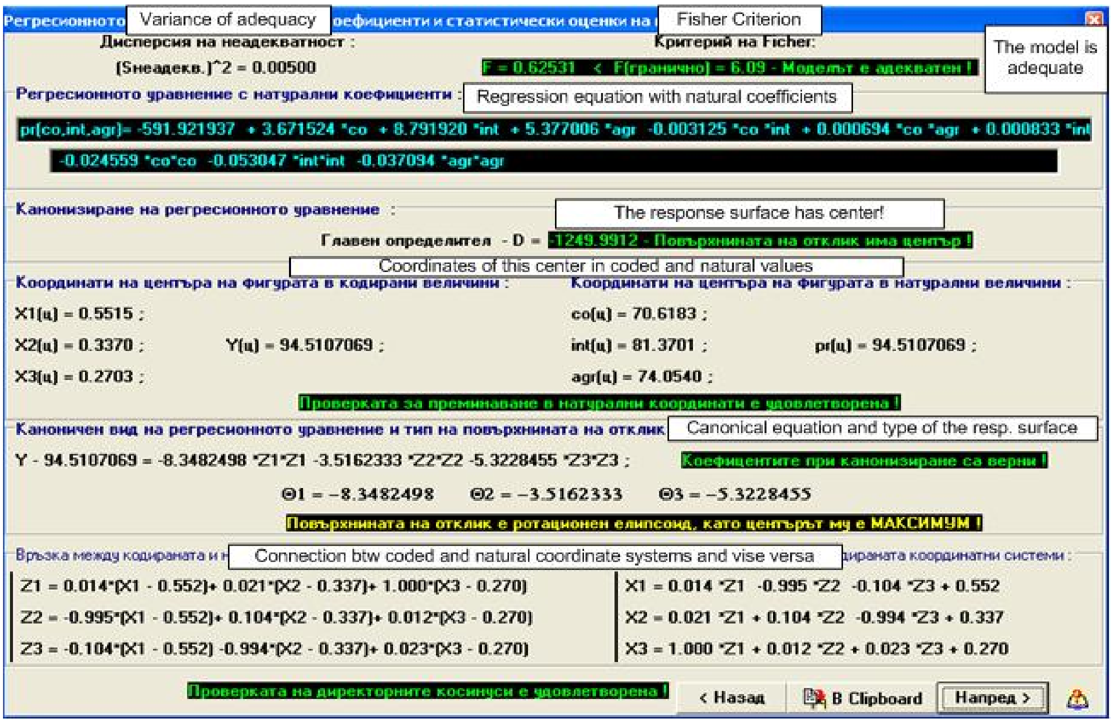



**Figure 53 Regression equation in natural form and statistical analysis of the model**

On the next Figure 54 we see the two-dimensional intersections for all the possibilities (-1, 0, 1) for each of the input factors ($X_1$, $X_2$, $X_3$). For every cut we have a 3D graphic in MATLAB on which we can see how exactly the response surface looks like and where exactly the maximum is.

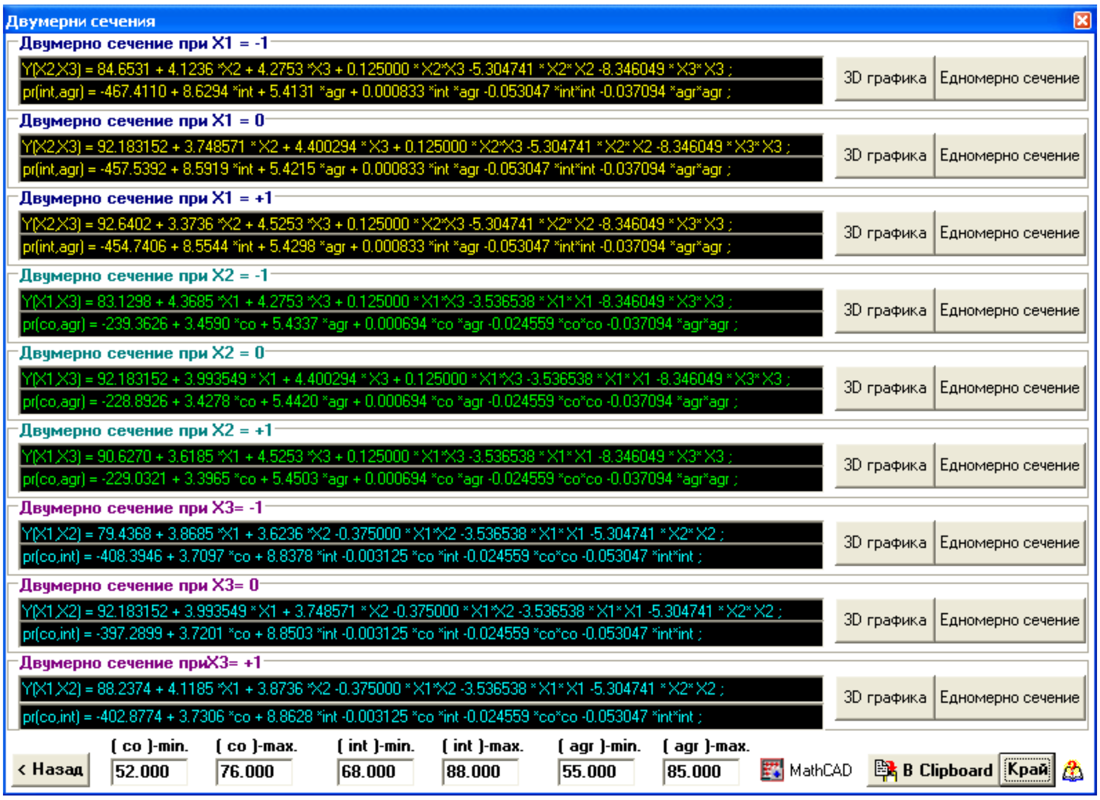



**Figure 54 Two-dimensional intersections**

As for the previous experiment by motivation of 75% we will not explain here the resultant graphics but we will show them in the Appendix. The explanations and the gained data are analogically to the first experiment by motivation of 55%, only the gained results are with higher values because of the higher motivation. For better understanding we are showing a comparison between the resultant data from the three experiments in the following table 32.

**Table 32 Comparison between the performance values by motivation of 55%, 70% and 85%, measured on the out lines of the factor space**

| Conscientious-ness [%] | Intelligence [%] | Agreeableness [%] | By Motivation 55% | By Motivation 70% | By Motivation 85% |
|---|---|---|---|---|---|
| | | | Performance | Performance | Performance |
| 52 | 60 | 40 | 7 | 8,2 | 16,6 |
| | | 75 | 42 | 50,3 | 57 |
| | | 80 | 40,7 | 49 | 55 |
| | | 100 | 20 | 25 | 30 |
| | | | | | |
| 52 | 60 | 40 | 7 | 8,2 | 16,6 |
| | 80 | | 29 | 33,7 | 40,7 |
| | 100 | | 13,3 | 15,3 | 22,3 |
| | | | | | |
| 76 | 60 | 40 | 13 | 16 | 25,4 |
| | | 80 | 49 | 58 | 65 |
| | | 100 | 29 | 34,6 | 39,6 |
| | | | | | |
| 76 | 60 | 40 | 13 | 16 | 25,4 |
| | 83 | | 34 | 40,4 | 47,7 |
| | 100 | | 17 | 21,2 | 28,2 |
| | | | | | |
| 40 | 68 | 40 | 8 | 9 | 16,6 |
| 55 | | | 22 | 26 | 34 |
| 70 | | | 27 | 32 | 40 |
| 80 | | | 24,4 | 29,4 | 38,2 |
| 90 | | | 18 | 22 | 31,3 |
| | | | | | |
| 40 | 88 | 40 | 15,3 | 17,5 | 24,5 |
| 68 | | | 33 | 39 | 46 |
| 90 | | | 23 | 28,4 | 36 |

The comparison between the performance values by motivation of 55%, 70% and 85% made on the out limits/lines of the factor space represents values from the figures 44, 45 and 46 and also from the rest figures in the Appendix for the other two experiments.

We can clearly see in the table that we have a significant increase in the performance values by motivation of 85% in comparison with the other two experiments. Anyway the dependencies btw the different characteristics and the productivity stay the same as already explained.

Because of this we won't give them once again but we will show only the differences between the maximum performance values, taken from the maximum point of the response surface (figures 42, 49 and 53) for every of the experiments. This can be seen below:

By motivation of 55% (figure 42):

| Conscientiousness = 70,19 %<br>Intelligence = 81,25 %<br>Agreeableness = 74,05 % | Performance = 71,35 % |
|---|---|

By motivation of 70% (figure 49):

| Conscientiousness = 70,47 %<br>Intelligence = 81,29 %<br>Agreeableness = 74,11 % | Performance = 84,67 % |
|---|---|

By motivation of 85% (figure 53):

| Conscientiousness = 70,6 %<br>Intelligence = 81,4 %<br>Agreeableness = 74,05 % | Performance = 94,5 % |
|---|---|

It is clear to see that the differences between the values of the psychological characteristics are imperceptible but we see significant difference in the Performance values. This can be explained with the enormous influence of the motivation over the working process. As we have seen in the very beginning the correlation value between motivation and performance is 0.968941, which is proved once again from the values above.

## 6.4 The developed model for IT human performance prediction

We will give a short summary over the achievements of the developed method:

- We were able to choose the most important human factors: Motivation; Conscientiousness; Intelligence and Agreeableness, on which to build our model.

- We have conducted three experiments by three special values of the motivation factor, because of the complex subjective-psychological dependence btw motivation and performance.

  The growing of the motivation up to 85% is connected with growing of the desire to give the best possible productiveness at work and by higher values than 85% this desire is decreasing. This can be explained with the fact that the people with 100% of motivation find it difficult to see perspective for development as they have already reached the maximum; this is a kind of de-motivation and results in lower performance levels. By 55%; 70% and 85% motivation can be observed a significant change in the performance values (figure 29) values and because of this we are designing our experiments by these special values.

- For the three experiments we have gained three statistically correct mathematical models as follows:

(6.25)

*Motivation of 55%*

Performance[%] = pr(co, int, agr) = -523.021607 + 3.139297*co + 7.793311*int + 4.525325*agr – 0.002677*co*int + 0.001674*co*agr – 0.021694*co*co – 0.046799*int*int – 0.031346*agr*agr

*Motivation of 70%*

Performance[%] = pr(co, int, agr) = -611.111026 + 3.619927*co + 9.066844*int + 5.388229*agr – 0.001979*co*int + 0.001319*co*agr – 0.025236*co*co – 0.054906*int*int – 0.036977*agr*agr

*Motivation of 85%*

Performance[%] = pr(co, int, agr) = -591.921937 + 3.671524*co + 8.791920*int + 5.377006*agr – 0.003125*co*int + 0.000694*co*agr + 0.000833*int – 0.024559*co*co – 0.053047*int*int – 0.037094*agr*agr

- *The connection btw agreeableness and performance:* the growing values of agreeableness up to 75% characterize with growing of the performance because the employee is able to communicate and cooperate with his colleagues, he is able to accept others' ideas and to follow instructions; after these values the person loses his own judgment and cannot resolve any problem alone. The software engineer agrees with everyone and is not able to take decisions anymore and this leads to low values of performance. When the values are low around 40%, he is not able to cooperate and works very difficult with other people and this of course means also low productivity.

- *The connection btw intelligence and performance:* with the increasing of the intelligence after a specific point (around 80%), the observed software team members start to take very complex decisions and don't choose the optimal algorithm for resolving a problem. This leads to complications and more mistakes in the work process, the employees need more time and the solutions are not optimal, because of this it is logical to observe the decreasing of the performance. By low values, even by 60% we have very low productiveness which shows that we need employees with intelligence over the average in order to manage the software engineering process.

- *The connection btw conscientiousness and performance:* by growing until 70% it shows that the software specialist is trying to do his best and to manage his work as good as possible. From other side this characteristic hinders the process of ignoring the unimportant details in the everyday work, and exactly this leads to decreasing of the performance, when the conscientiousness is higher than 70%. The employee loses too much time in checking details and spending time for not so important problems which needs more time and results into lower productivity. When the values are low, until 55% we have very low performance and this is to be explained with the fact that such employees are not doing their job with the needed respect and cautious.

- The results of the whole development process of the predictive model can be characterized in the following manner:

$$ {}^{eval(BigFive)}_{DoE}HF^{FMEA}_{SoftwareProcess} = \mathcal{F}(Motivation,\ Conscientiousness,\ Openness,\ Agreeableness) \quad (6.26) $$

The following figure summarizes the characteristics qualification of the IT human factors for their high performance in software development teams and structures.

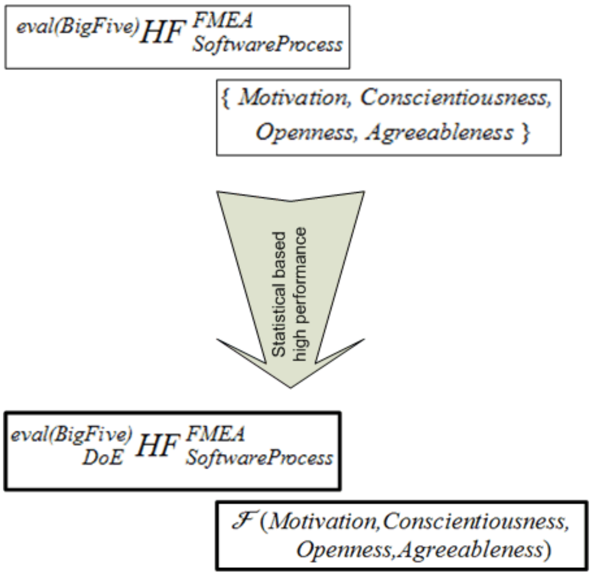



**Figure 55** Quantified IT human factors for high performance

The developed mathematical model gives the possibility to predict the productiveness of the examined person based on his/her special psychological traits. This supports the process of IT personnel recruitment and also the whole process of IT personnel development with a powerful tool for achieving of better software quality.

## 6.5 Summary over the development of the predictive model

1) Based on the statistical analysis with which Chapter 5 ends, we have found out the connection (correlation) btw the following complex psychological characteristics and the performance.

1. Openness
2. Conscientiousness
3. Extroversion
4. Agreeableness
5. Neuroticism
6. Experience
7. Motivation

Having the correlation analysis we were able to decide that the Motivation; Conscientiousness; Intelligence and Agreeableness are the most influencing factors that we are going to investigate.

2) We have decided to use the Design of Experiment method for the modelling and to build a rotatable experiment because of the following advantages (Khuri & Cornell 1996) (Myers, Montgomery & Cook 2009):

- Gain maximum information from a specified number of experiments;
- Study effects individually by varying all operating parameters simultaneously;
- Take account of variability in experiments or processes themselves;
- Characterize acceptable ranges of key and critical process parameters contributing to identification of a design space, which helps to provide an "assurance of quality;
- Guarantees the invariance of the plan and of the parameter of optimization;
- The model obtained by the rotatable plan describes the response surface with equal accuracy (equal variance) in all directions of the coordinate axes;
- In the whole factors space, the parameter of optimization has the same variance. This assures that the calculation accuracy of the optimization parameter is independent from the place where we are going to build the experiment;
- The variance of the optimization parameter does not change by rotation and translation of it coordination system. This allows us to conduct the canonical experiment (rotation and translation of the coordination system) with the goal to find the geometrical figure of the designed experiment;
- The fact that by rotatable experiments the variance does not change assures the correctness of the statistical analysis of the gained mathematical model.

3) The factorial space according to the values of the input factors has been determined. The input data for the experiment have been prepared. On figures 38 and 39 can be seen the input, where the factor space is determined from the values of: Conscientiousness from 38% to 90%; Intelligence from 58% to 98%; Agreeableness from 40% to 100%.

4) The full matrix of the planned experiment by motivation of 55% (figure 40) is build and the concrete performance values have been measured. The same have also been done by motivation of 70% and 85% (figures 47 and 51). On the figures (41, 48 and 52) can be seen the calculation of the coefficients of the mathematical model and after this (figures 42, 49 and 53) the statistical evaluation for correctness of the models and the regression equations as end result.

5) When having the mathematical models with all important coefficients, there have been made statistical checks (figures 42, 49, 53) if they are adequate. They show that all the models are adequate and this means that we can proceed with the next step, the analysis of

the two-dimensional intersections (figures 43, 50 and 54) of the response surface.

6) It has been done a canonical analysis of all the models (figures 42, 49 and 53) in order to determine the geometrical kind of the response surfaces. It is in all three cases a rotational ellipsoid and the center is its maximum or we have in the center maximum performance.

7) We have build three experiments by three special values of the *motivation factor*, because of the complex subjective-psychological dependence btw motivation and performance. The growing of the motivation up to 85% is connected with growing of the desire to give the best possible productiveness at work and by higher values than 85% this desire is decreasing. This can be explained with the fact that the people with 100% of motivation find it difficult to see perspective for development as they have already reached the maximum; this is a kind of de-motivation and results in lower performance levels. By 55%; 70% and 85% motivation can be observed a significant change in the performance values (figure 29) values and because of this we are designing our experiments by these special values.

8) For the three experiments we have gained three statistically correct mathematical models as follows:

(6.25)

*Motivation of 55%*
Performance[%] = pr(co, int, agr) = -523.021607 + 3.139297*co + 7.793311*int + 4.525325*agr – 0.002677*co*int + 0.001674*co*agr – 0.021694*co*co – 0.046799*int*int – 0.031346*agr*agr

*Motivation of 70%*
Performance[%] = pr(co, int, agr) = -611.111026 + 3.619927*co + 9.066844*int + 5.388229*agr – 0.001979*co*int + 0.001319*co*agr – 0.025236*co*co – 0.054906*int*int – 0.036977*agr*agr

*Motivation of 85%*
Performance[%] = pr(co, int, agr) = -591.921937 + 3.671524*co + 8.791920*int + 5.377006*agr – 0.003125*co*int + 0.000694*co*agr+0.000833*int – 0.024559*co*co – 0.053047*int*int – 0.037094*agr*agr

9) Comparing the three prognostic models we see that the response surfaces (rotatable ellipsoids) and the mathematical equations are identical. The differences are only in the concrete values and we are going to show the maximum values for each ellipsoid:

By motivation of 55% we have maximum performance of 71,35% (figure 42).

By motivation of 70% we have maximum performance of 84,67% (figure 49).

By motivation of 85% we have maximum performance of 94,51% (figure 53).

It is clear that with higher motivation we have also higher productiveness.

10) The connection btw the other three input factors and the performance are explained in the following manner:

- The connection btw *agreeableness* and performance is an ellipse (figure 44), the growing values of agreeableness up to 75% characterize with growing of the performance because the employee is able to communicate and cooperate with his colleagues, he is able to accept others' ideas and to follow instructions; after these values the person loses his own judgment and cannot resolve any problem alone. The software engineer agrees with everyone and is not able to take decisions anymore and this leads to low values of performance. When the values are low around 40%, he is not able to cooperate and works very difficult with other people and this of course means also low productivity.

- The connection btw *intelligence* and performance is also an ellipse (figure 44). The observations have shown that with the increasing of the intelligence after a specific point (around 80%), the observed software team members start to take very complex decisions and don't choose the optimal algorithm for resolving a problem. This leads to complications and more mistakes in the work process, the employees need more time and the solutions are not optimal, because of this it is logical to observe the decreasing of the performance. By low values, even by 60% we have very low productiveness which shows that we need employees with intelligence over the average in order to manage the software engineering process.

- The connection btw *conscientiousness* and performance is also an ellipse (figure 45). This can be explained with the specific of this psychological characteristic and it is that by growing until 70% it means that the software specialist is trying to do his best and to manage his work as good as possible. From other side this characteristic hinders the process of ignoring the unimportant details in the everyday work, and exactly this leads to decreasing of the performance, when the conscientiousness is higher than 70%. The employee loses too much time in checking details and spending time for not so important problems which needs more time and results into lower productivity. When the values are low, until 55% we have very low performance and this is to be explained with the fact that such employees are not doing their job with the needed respect and cautious.

11) Figures 43, 50 and 54 show the two-dimensional intersections which are used for the visualization of the response surface of the performance. We have shown there figures for performance by motivation of 55% - figures 44, 45 and 46. In this way we are able to give geometrical interpretation of the gained models and to find the dependencies btw the performance and the three specific psychological features. The additional figures for the other two experiments are shown in the Appendix.

12) For better understanding we have also additional one-dimensional intersections, on which can be seen concrete values by different factors combinations, but as this is additional information, it is shown in the Appendix.

13) The results of our experiment based on the DoE method could be characterized in the following short manner as:

$$ {}^{eval(BigFive)}_{DoE}HF^{FMEA}_{SoftwareProcess} = \mathcal{F}(Motivation,\ Conscientiousness,\ Openness,\ Agreeableness) \quad (6.26) $$

# 7 Chapter – Experimental validation of the predictive model for IT human performance

In the last chapter are shown real examples of the effectiveness of the developed mathematical model. We have developed also a special web-application which realizes the test and after this transforms the gained information into input data for our model and ends with the predicted productiveness for the examined person. The gained statistical information shows the accurateness of the method and proves its positive use for improving the software development process in the way that we can choose more reliable and productive personnel.

## 7.1 The actual application of the model

Here we are going to prove the adequacy and the effectiveness of the gained prognostic mathematical models (Georgieva et al, 2011 c). This has been done with the conduction of many surveys in German and Bulgarian software companies.

As we have seen until the moment we have designed a complex mathematical model that describes the human productivity in the software development field based on the individual personal characteristics. We will show once again the three equations according to the measured motivation and after this we will give concrete real examples that show the correctness of the model.

*Motivation of 55%*

Performance[%] = pr(co, int, agr) = -523.021607 + 3.139297*co + 7.793311*int + 4.525325*agr – 0.002677*co*int + 0.001674*co*agr – 0.021694*co*co – 0.046799*int*int – 0.031346*agr*agr (7.1)

*Motivation of 70%*

Performance[%] = pr(co, int, agr) = -611.111026 + 3.619927*co + 9.066844*int + 5.388229*agr – 0.001979*co*int + 0.001319*co*agr – 0.025236*co*co – 0.054906*int*int – 0.036977*agr*agr (7.2)

*Motivation of 85%*

Performance[%] = pr(co, int, agr) = -591.921937 + 3.671524*co + 8.791920*int + 5.377006*agr – 0.003125*co*int + 0.000694*co*agr+0.000833*int – 0.024559*co*co – 0.053047*int*int – 0.037094*agr*agr (7.3)

### 7.1.1 Examples

The following examples are a mean representative of the gained questionnaire data. We have showed the data in the form of 12 examples (case studies), but actually they are summarizing the data from 50 questioned software employees from different companies.

***Real case 1:***

84% agreeableness;
92% conscientiousness;
76% intellect;
Motivation 75%;
Estimated productivity **70%.**

Productivity (calculated from the model) = -611.111026 + 333.033284 + 689.080144 + 452.611236 – 13.837168 + 10.193232 – 213.597504 – 317.137056 – 260.909712 = **68.32543%**
Difference = 1.68%

***Real case 2:***

92% agreeableness;
84% conscientiousness;
70% intellect;
Motivation 85%;
Estimated productivity **70%.**

Productivity (calculated from the model) = -591.921937 + 308.408016 + 615.4344 + 494.684552 – 18.375 + 5.363232 + 0.05831 – 173.288304 – 259.9303 – 313.963616 = **66.469353%**
Difference = 3.53%

***Real case 3:***

82% agreeableness;
90% conscientiousness;
86% intellect;
Motivation 85%;
Estimated productivity **80%.**

Productivity (calculated from the model) = -591.921937 + 330.43716 + 756.10512 + 440.914492 – 24.1875 + 5.12172 + 0.071638 – 198.9279 – 392.335612 – 249.420056 = **75.857125%**
Difference = 4.14%

***Real case 4:***

90% agreeableness;
58% conscientiousness;
92% intellect;
Motivation 55%;
Estimated productivity **58%.**

Productivity (calculated from the model) = -523.021607 + 182.079226 + 716.984612 + 407.27925 – 14.284472 + 8.73828 -72.978616 – 396.106736 – 253.9026 = **54.787337**
Difference = 3.22%

***Real case 5:***

82% agreeableness;
68% conscientiousness;
94% intellect;

Motivation 75%;
Estimated productivity **75%.**

Productivity (calculated from the model) = -611.111026 + 246.155036 + 852.28336 + 441.834778 – 12.649768 + 7.354744 – 116.691264 – 485.149416 – 248.633348 = **73.393096%**
Difference = 1.61%

***Real case 6:***

82% agreeableness;
74% conscientiousness;
82% intellect;
Motivation 75%;
Estimated productivity **85%.**

Productivity (calculated from the model) = -611.111026 + 267.874598 + 743.481208 + 441.834778 – 12.008572 + 8.003692 – 138.192336 – 369.187944 – 248.633348 = **82.06105%**
Difference = 2.94%

***Real case 7:***

94% agreeableness;
90% conscientiousness;
96% intellect;
Motivation 85%;
Estimated productivity **55%.**

Productivity (calculated from the model) = -591.921937 + 330.43716 + 844.02432+ 505.438564 – 27 + 5.87124 + 0.079968 – 198.9279 – 488.881152 – 327.762584 = **51.357679%**
Difference = 3.65%

***Real case 8:***

72% agreeableness;
72% conscientiousness;
78% intellect;
Motivation 85%;
Estimated productivity **90%**.

Productivity (calculated from the model) = -591.921937 + 264.349728 + 685.76976 + 387.144432 – 17.55 + 3.597696 + 0.064974 – 127.313856 – 322.737948 – 192.295296 = **89.107553%**
Difference = 0.9%

***Real case 9:***

82% agreeableness;
66% conscientiousness;
96% intellect;
Motivation 85%;
Estimated productivity **75%.**

Productivity (calculated from the model) = -591.921937 + 242.320584 + 844.02432 + 440.914492 - 19.8 + 3.755982 + 0.079968 - 106.979004 - 488.881152 - 249.420056 = **74.093197%**
Difference = 0.91%

***Real case 10:***

96% agreeableness;
90% conscientiousness;
88% intellect;
Motivation 55%;
Estimated productivity **50%.**

Productivity (calculated from the model) = - 523.021607 + 282.53673 + 685.811368 + 434.4312 - 21.20184 + 14.46336 - 175.214 - 362.411456 - 288.884736 = **46.508915%**
Difference = 3.5%

***Real case 11:***

90% agreeableness;
62% conscientiousness;
66% intellect;
Motivation 85%;
Estimated productivity **70%.**

Productivity (calculated from the model) = - 591.921937 + 227.634488 + 580.26672 + 483.93054 - 12.7875 + 3.87252 + 0.054978 - 94.404796 - 231.072732 - 300.4614 = **65.110881%**
Difference = 4.89%

***Real case 12:***

94% agreeableness;
62% conscientiousness;
94% intellect;
Motivation 70%;
Estimated productivity **60%.**

Productivity (calculated from the model) = - 611.111026 + 224.435474 + 852.28336 + 506.493526 - 11.533612 + 7.687132 - 97.007184 - 485.149416 - 326.728772 = **59.369482%**
Difference = 0.64%

We can summarize that the difference between Estimated and Calculated Productivity is not bigger than 5%, which is a very important proof for the correctness of the developed model. We will observe in the next point a statistical analysis of 100 additional real examples, which shows once again the adequacy and efficiency of our prognostic mathematical model.

## 7.2 The Software Human Factors Test web application

In order to automate the questioning process and the processing of the gained data into actual results about a concrete person and also to show once again the effectiveness and correctness of the developed method we have developed a web application that conducts the explained actions and supports us with the final results.

Here we will describe the test tool and we will present screenshots with different results. This application is for us from great help because it enables the test-quiz and after this the evaluation of the results and their use in the already explained formulas (the mathematical model) that describe the personal productivity. In this way we end with the concrete performance for every tested person and we can also observe the whole statistic of the people that have already done the test.

Let us now start with the first screenshot (Figure 56) of the tool: when loading the home page the user is presented with the option to start a new test, to resume an unfinished one or to view the results of the own completed test and also to view the whole statistics for all the completed tests.

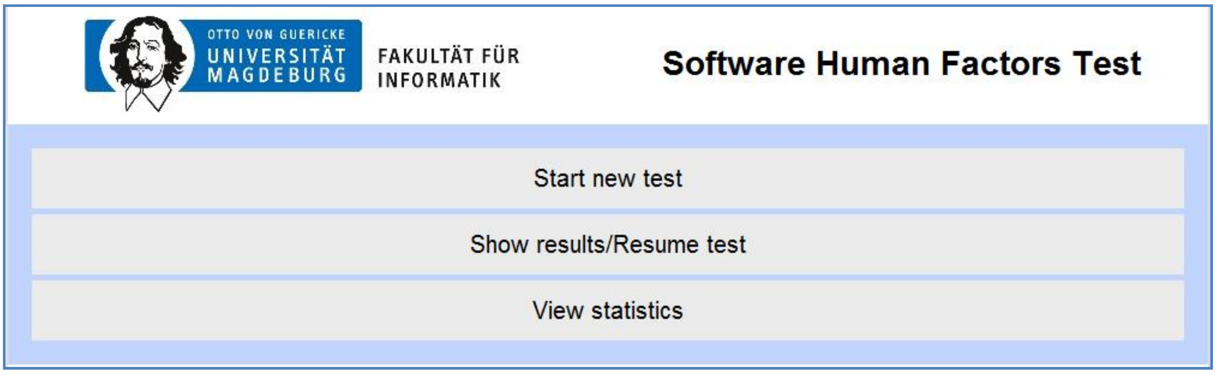



**Figure 56 Home page of the web-application**

If the user decides to start a new test he/she is brought to a page with the test questions in a shuffled order, which looks like the following.


**Figure 57 Web-application quiz page**

Most of the questions have 5 possible answers – very accurate, accurate, inaccurate, very inaccurate and other. When the last is selected a textbox is displayed where the user can enter a custom textual answer. Some of the questions are answered only by true or false and some need to be answered by some text. We can see these different types of questions on the next Figure 58. More detailed explanation about the different types of questions and answers have been given in Chapter 5.

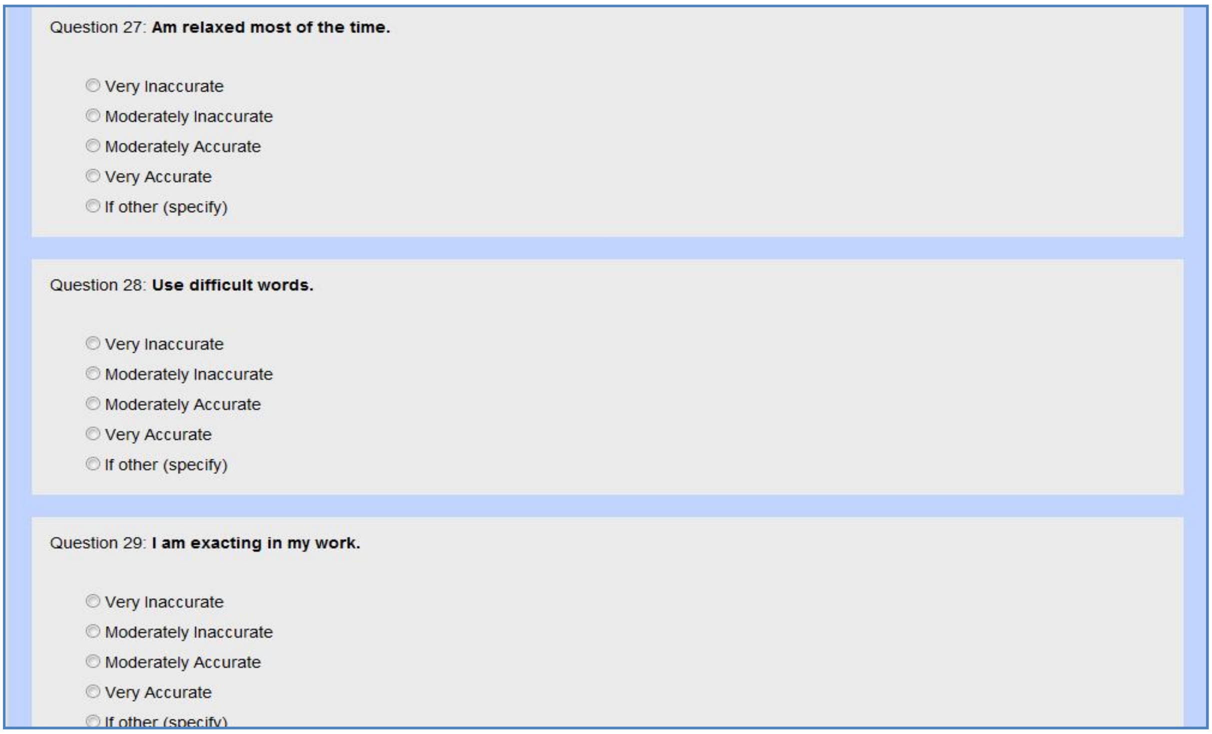



**Figure 58 Web-application's question types**

At the end of the test page, the user can click Submit which will save the answers. This can be done even if the quiz is not completed.

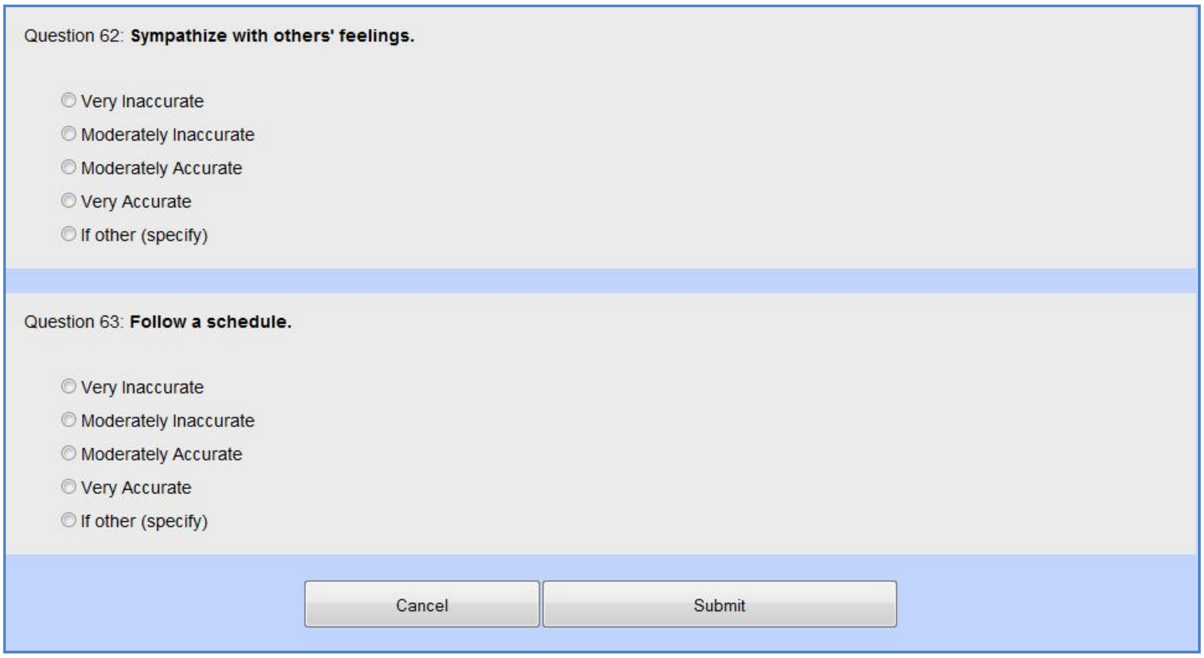



**Figure 59 The end of the quiz page**

The user is then redirected to a page showing the ID of the taken test. From there, if the test is not completed it can be resumed, and if it is completed – the results can be viewed. The user can also go back to the home page.

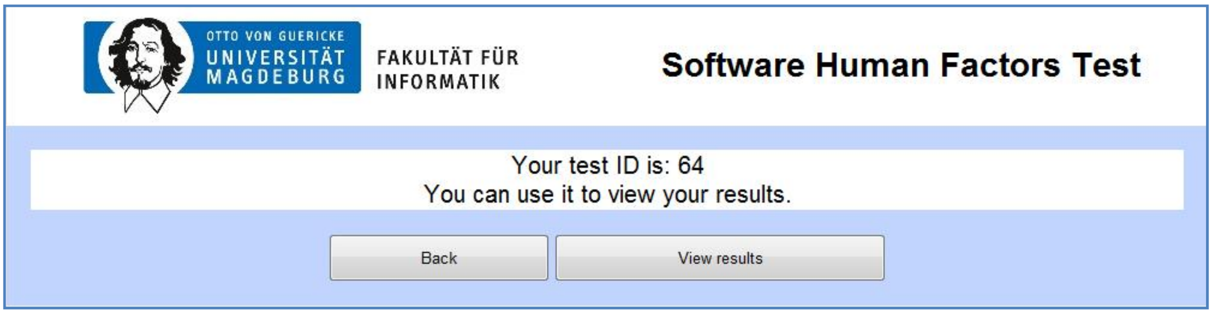



**Figure 60 The web - application's Quiz Finished Page**

From the home page the user can again resume an unfinished test or view the results of a finished one by clicking the button "Show results/Resume test", using the ID of his personal test.

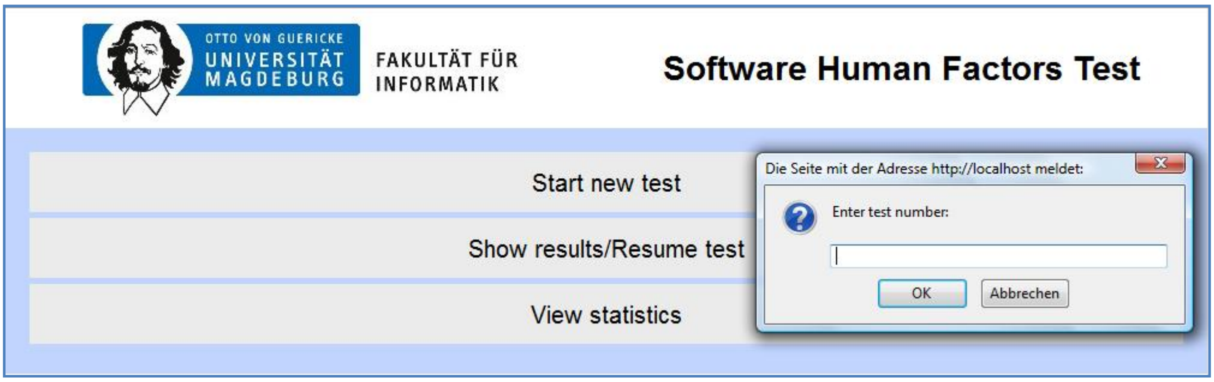


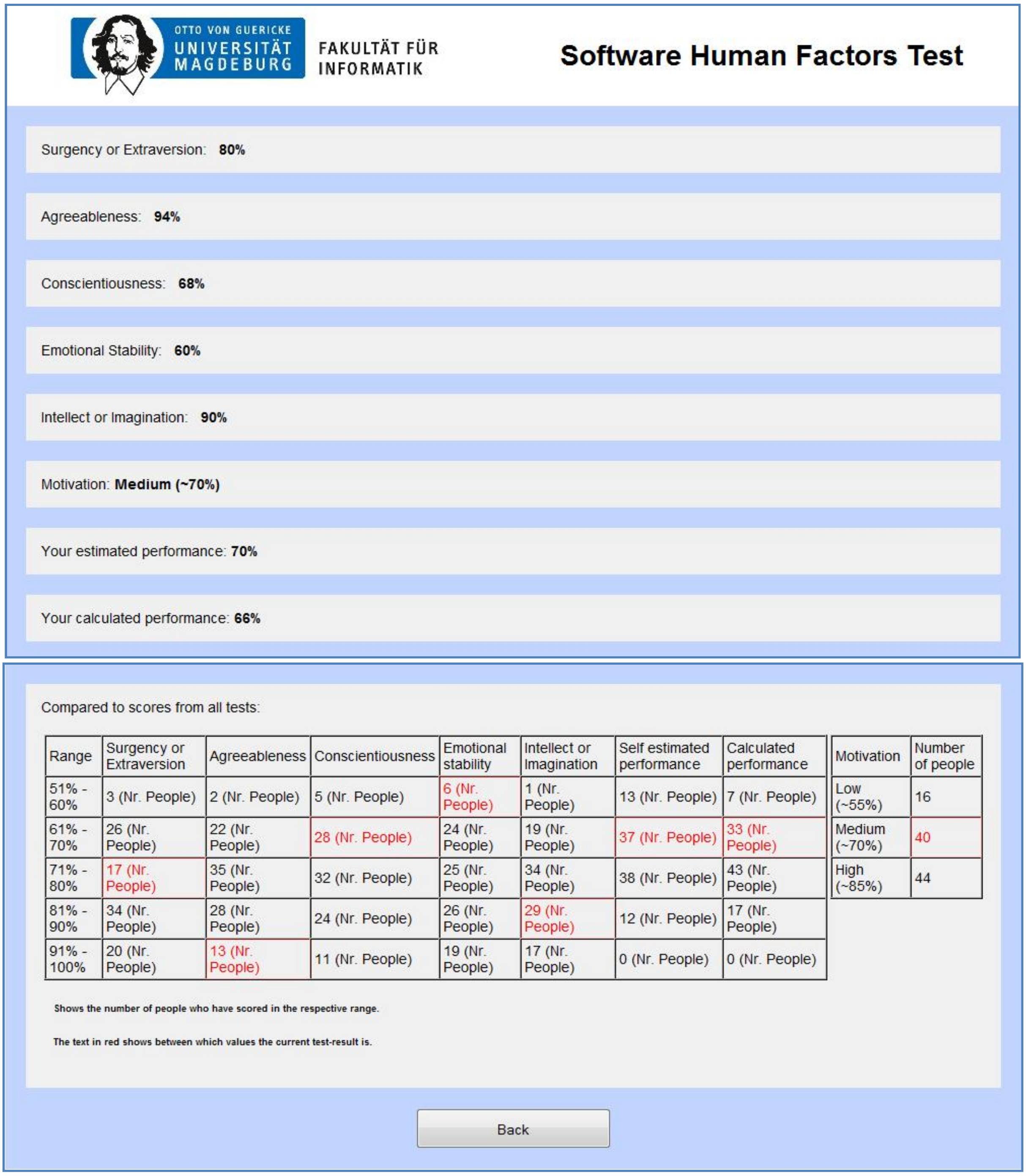

**Figure 61 Screenshot of the application's popup when "Show results/Resume Test" button is clicked**

Clicking that button pops up a field where the user is required to enter the ID of the test he/she wants to resume or view the results of.

If a test with the entered ID does not exist, the user is redirected to a page with a message that tells this. From there he/she can go back to the home page. This could be seen on the next Figure 62.


**Figure 62 Window shown, when the test does not exist**

Resuming an unfinished test loads the quiz page with the questions in the order they were when the test was created. Also the answers are recreated so if the user wants he/she can change them before finishing the test.
Viewing results brings the user to the following page (Figure 63) showing his/her score for the five measured factors, the self estimated performance and motivation and the calculated performance by the developed mathematical model. There is also a table with statistics for all completed tests. The columns in the table represent each factor and the rows – a range of scores. The cells in the table show how much people have scored a value in the respective range for the corresponding factor. The text in red shows between which values the current test-result is.


FAKULTÄT FÜR INFORMATIK

Software Human Factors Test

Surgency or Extraversion: **80%**

Agreeableness: **94%**

Conscientiousness: **68%**

Emotional Stability: **60%**

Intellect or Imagination: **90%**

Motivation: **Medium (~70%)**

Your estimated performance: **70%**

Your calculated performance: **66%**

Compared to scores from all tests:

| Range | Surgency or Extraversion | Agreeableness | Conscientiousness | Emotional stability | Intellect or Imagination | Self estimated performance | Calculated performance |
|---|---|---|---|---|---|---|---|
| 51% - 60% | 3 (Nr. People) | 2 (Nr. People) | 5 (Nr. People) | 6 (Nr. People) | 1 (Nr. People) | 13 (Nr. People) | 7 (Nr. People) |
| 61% - 70% | 26 (Nr. People) | 22 (Nr. People) | 28 (Nr. People) | 24 (Nr. People) | 19 (Nr. People) | 37 (Nr. People) | 33 (Nr. People) |
| 71% - 80% | 17 (Nr. People) | 35 (Nr. People) | 32 (Nr. People) | 25 (Nr. People) | 34 (Nr. People) | 38 (Nr. People) | 43 (Nr. People) |
| 81% - 90% | 34 (Nr. People) | 28 (Nr. People) | 24 (Nr. People) | 26 (Nr. People) | 29 (Nr. People) | 12 (Nr. People) | 17 (Nr. People) |
| 91% - 100% | 20 (Nr. People) | 13 (Nr. People) | 11 (Nr. People) | 19 (Nr. People) | 17 (Nr. People) | 0 (Nr. People) | 0 (Nr. People) |

| Motivation | Number of people |
|---|---|
| Low (~55%) | 16 |
| Medium (~70%) | 40 |
| High (~85%) | 44 |

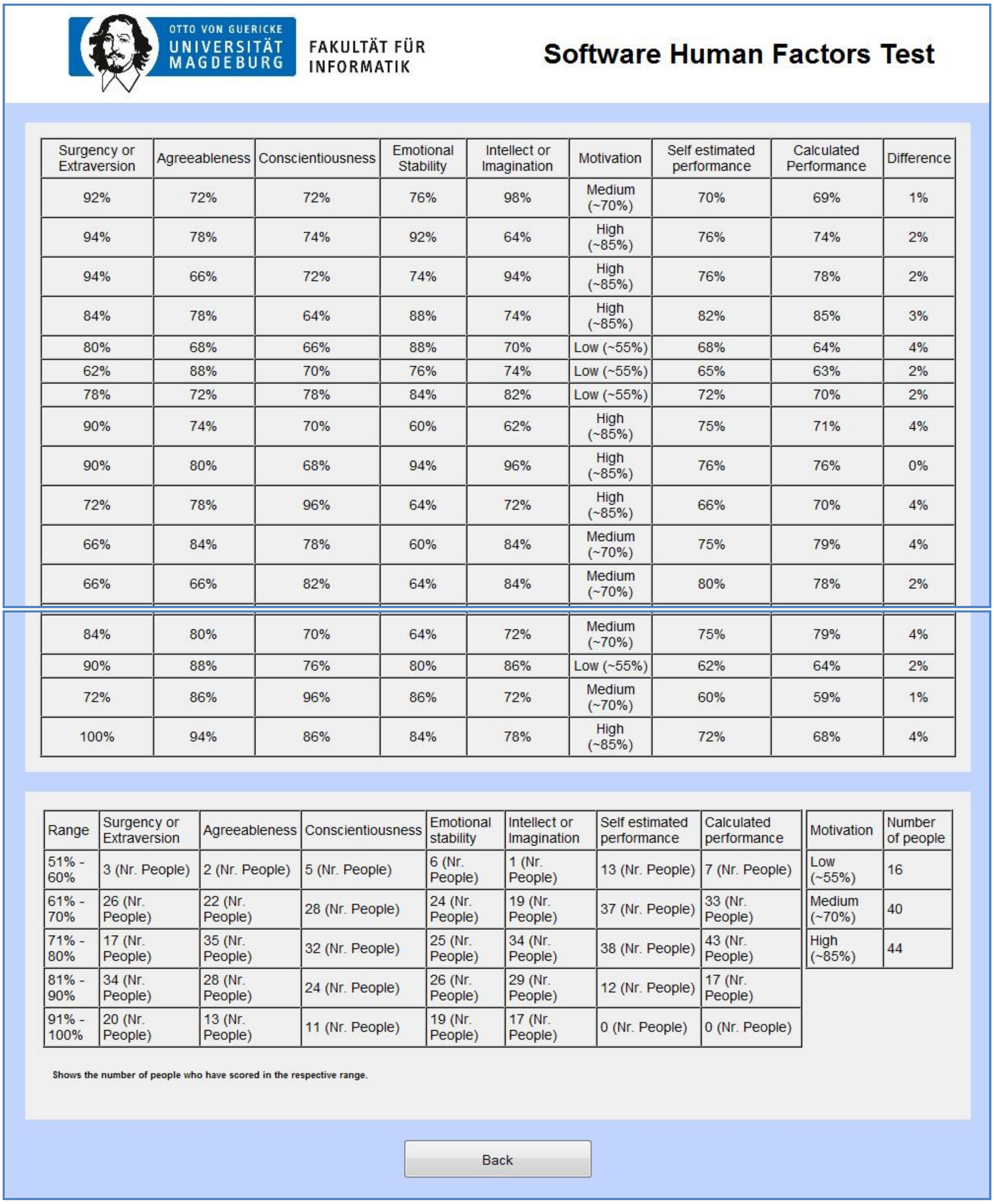
Shows the number of people who have scored in the respective range.

The text in red shows between which values the current test-result is.

Back

**Figure 63 The Test Results Page for a concrete person**

On the result page can be also observed the values for Extraversion and Emotional Stability. Although that they don't take part in the calculation of the performance values, they are displayed for additional information of the test-taker and for completeness of the questionnaire.




| Surgency or Extraversion | Agreeableness | Conscientiousness | Emotional Stability | Intellect or Imagination | Motivation | Self estimated performance | Calculated Performance | Difference |
|---|---|---|---|---|---|---|---|---|
| 92% | 72% | 72% | 76% | 98% | Medium (~70%) | 70% | 69% | 1% |
| 94% | 78% | 74% | 92% | 64% | High (~85%) | 76% | 74% | 2% |
| 94% | 66% | 72% | 74% | 94% | High (~85%) | 76% | 78% | 2% |
| 84% | 78% | 64% | 88% | 74% | High (~85%) | 82% | 85% | 3% |
| 80% | 68% | 66% | 88% | 70% | Low (~55%) | 68% | 64% | 4% |
| 62% | 88% | 70% | 76% | 74% | Low (~55%) | 65% | 63% | 2% |
| 78% | 72% | 78% | 84% | 82% | Low (~55%) | 72% | 70% | 2% |
| 90% | 74% | 70% | 60% | 62% | High (~85%) | 75% | 71% | 4% |
| 90% | 80% | 68% | 94% | 96% | High (~85%) | 76% | 76% | 0% |
| 72% | 78% | 96% | 64% | 72% | High (~85%) | 66% | 70% | 4% |
| 66% | 84% | 78% | 60% | 84% | Medium (~70%) | 75% | 79% | 4% |
| 66% | 66% | 82% | 64% | 84% | Medium (~70%) | 80% | 78% | 2% |
| 84% | 80% | 70% | 64% | 72% | Medium (~70%) | 75% | 79% | 4% |
| 90% | 88% | 76% | 80% | 86% | Low (~55%) | 62% | 64% | 2% |
| 72% | 86% | 96% | 86% | 72% | Medium (~70%) | 60% | 59% | 1% |
| 100% | 94% | 86% | 84% | 78% | High (~85%) | 72% | 68% | 4% |

| Range | Surgency or Extraversion | Agreeableness | Conscientiousness | Emotional stability | Intellect or Imagination | Self estimated performance | Calculated performance |
|---|---|---|---|---|---|---|---|
| 51% - 60% | 3 (Nr. People) | 2 (Nr. People) | 5 (Nr. People) | 6 (Nr. People) | 1 (Nr. People) | 13 (Nr. People) | 7 (Nr. People) |
| 61% - 70% | 26 (Nr. People) | 22 (Nr. People) | 28 (Nr. People) | 24 (Nr. People) | 19 (Nr. People) | 37 (Nr. People) | 33 (Nr. People) |
| 71% - 80% | 17 (Nr. People) | 35 (Nr. People) | 32 (Nr. People) | 25 (Nr. People) | 34 (Nr. People) | 38 (Nr. People) | 43 (Nr. People) |
| 81% - 90% | 34 (Nr. People) | 28 (Nr. People) | 24 (Nr. People) | 26 (Nr. People) | 29 (Nr. People) | 12 (Nr. People) | 17 (Nr. People) |
| 91% - 100% | 20 (Nr. People) | 13 (Nr. People) | 11 (Nr. People) | 19 (Nr. People) | 17 (Nr. People) | 0 (Nr. People) | 0 (Nr. People) |

| Motivation | Number of people |
|---|---|
| Low (~55%) | 16 |
| Medium (~70%) | 40 |
| High (~85%) | 44 |

**Figure 64 The Software Human Factors Test Statistics Page**

Going back to the home page, the user can see the entire statistics (figure 64) of all already taken tests. They are shown in a separate page in a shuffled manner and without the test IDs, so that no one can connect a particular ID with the shown statistics. We can see also a table similar to the one in the results page, but since the statistics is global, it is not matched to any specific test. We can see the percentage range of each psychological feature and the number of the people that belong to it.

### 7.2.1 Analysis of the gained information

The developed web-application 'Software Human Factors Test' was given to a number of software companies in order to gain real results and to observe the correctness of the developed prognostic method. We have gained exactly 100 useful test results, a part of which can be seen on Figure 64 and on the following Table 33. The complete list can be seen in the Appendix.

The data that we have collected shows that the developed mathematical model really predicts the human performance very accurate, the differences between the estimated and the calculated performance are not bigger than 5%, which is the confidence interval and this means that the method works very accurate and can be applied in the praxis without any doubts.

**Table 33 Part of the received employee information**

| Agreeableness | Conscientiousness | Intellect (Openness) | Motivation | Estimated performance | Calculated Performance | Difference |
|---|---|---|---|---|---|---|
| 88% | 82% | 70% | High (~85%) | 78% | 73% | 5% |
| 68% | 74% | 86% | High (~85%) | 86% | 87% | 1% |
| 82% | 86% | 66% | High (~85%) | 72% | 70% | 2% |
| 82% | 62% | 92% | High (~85%) | 75% | 78% | 3% |
| 92% | 88% | 76% | High (~85%) | 65% | 68% | 3% |
| 92% | 64% | 80% | High (~85%) | 78% | 75% | 3% |
| 72% | 94% | 68% | High (~85%) | 64% | 68% | 4% |
| 80% | 60% | 68% | Low (~55%) | 60% | 59% | 1% |
| 76% | 86% | 72% | Low (~55%) | 60% | 62% | 2% |
| 70% | 60% | 78% | Medium (~70%) | 85% | 81% | 4% |
| 90% | 64% | 74% | Medium (~70%) | 70% | 71% | 1% |
| 92% | 62% | 92% | Medium (~70%) | 62% | 65% | 3% |
| 88% | 70% | 74% | Low (~55%) | 65% | 63% | 2% |
| 86% | 74% | 60% | Medium (~70%) | 56% | 54% | 2% |
| 82% | 84% | 94% | Low (~55%) | 60% | 57% | 3% |
| 74% | 94% | 74% | Low (~55%) | 60% | 57% | 3% |
| 76% | 82% | 76% | Medium (~70%) | 82% | 80% | 2% |

The analysis of the information from Table 33 shows that the developed prognostic model about the influence of Motivation, Agreeableness, Conscientiousness and Intelligence over the Performance is adequate. The experimental testing of the method in real environment shows minimal mistake or difference of 5%, which allows us to claim that it works correct and can be used in the real process of employee' evaluation.

'Software Human Factors Test' is a reliable tool for productivity assessment in the software engineering field, which can be used by individuals and companies. Our research showed that it is an adequate source for performance assessment and in the same time provides the users with a good insight on the factors affecting their performance so that they know what they need to work on. The test can be used in addition to an interview for a job or as an addition to a set of some proven methods for improving productivity like Personal and Team Software Process and Capability Maturity Model Integration.

## 7.3 Summary over the experimental model validation

1) There have been conducted real case studies in different companies in order to build first ideas about the validation and effectiveness of the developed method for performance evaluation. These real examples showed that the difference between Estimated and Calculated Productivity is not bigger than 5%, which is a very important proof for the correctness of the developed model.

2) In order to automate the questioning process and the processing of the gained data into actual results about a concrete person and also to show once again the effectiveness and correctness of the developed method we have developed a web application that conducts the explained actions and supports us with the final results.

3) The developed web-application 'Software Human Factors Test' was given to a number of software companies in order to gain real results and to observe the correctness of the developed prognostic method. We have 100 useful test results, a part of which can be seen on Figure 64, in Table 33 and the complete list can be seen in the Appendix.

4) The observation of the collected data shows that the mathematical model really predicts the human performance very accurate, the differences btw the estimated and the calculated performance are not bigger than 5%, which means that the method works very accurate and can be applied in the praxis without any doubts.

5) 'Software Human Factors Test' is a reliable tool for productivity assessment in the software engineering field, which can be used by individuals and companies. Our research showed that it is an adequate source for performance assessment and in the same time provides the users with information about their personal factors affecting the performance. In this way they can use the tool also for self-evaluation and for further own development. The test can be used in addition to an interview for a job or as an addition to a set of some proven methods for improving productivity like PSP, TSP and CMMI.

6) The validated results *of high performance IT human factors* could be characterized as

$$ {}^{eval(BigFive)}_{eval(DoE)} HF^{FMEA}_{SoftwareProcess} = PERF\,(Motivation,\ Conscientiousness,\ Openness,\ Agreeableness) \qquad (7.4)$$

# Conclusion and Future Work

The PhD thesis starts with a big analysis of the existing methods for Risk Assessment with special focus over the Human Factors in them. The conducted literature review showed that the existing methods don't consider the human being as a factor responsible for different risks in the software engineering process and in this way influencing the end performance.

The second point in the research was to look from the other side. We have looked for psychological methods that measure and evaluate the influence of the personality over the software engineering process. We have discovered that such methods, at least in the software development do not exist but the conducted overview over the human factors in the software process has showed different perspectives:

- Slips and mistakes occurring in everyday human work including their base.
- Malfunctions and their relation to the behavioral model of the human being with regards to performing or not a certain task.
- Clearly recognized connection between emotions and risk behavior and different stressors influencing the people.
- Different levels of failures and factors that influence the human actions.
- Frameworks and taxonomies listing all different personal characteristics that influence the working process.

This observation was the major motivation for us to decide that there is an urgent need to develop such a method that will be able based on the specific psychological characteristics to prognosticate/evaluate the IT productivity for a special person.

The following analysis over the basic IT roles delivered us with the following description of the roles' most important competencies, which we have used in the further research.

$HF_{ProjectManager}$ = {*communicative, managerial skills, disciplined, respects the others, resolves conflicts, open minded, willing to develop himself, well-organized, goal-oriented, seeks improvement*}

$HF_{TeamLeader}$ = {*plan and prioritize the work, reviews team progress, flexible and adaptable, communicative, an effective advocate for the team, ability to lead and to impress*}

$HF_{BusinessAnalyst}$ = {*communicative, conceptual thinking, creativity, strategic and business thinking, problem solving, negotiation and decision making, customer oriented, team player*}

$HF_{SoftwareArchitect}$ = {*good decision maker, team player, performance oriented, technical understanding that supports the team, optimizing abilities, seeks new knowledge*}

$HF_{SoftwareDeveloper}$ = {*creativity, team player, tolerant, always in a learning mode, able to articulate own thoughts, respects others' ideas, structured thinking*}

$HF_{SoftwareTester}$ = {*creativity, flexibility, communicative, open-minded, respects the others*}

$HF_{QualityEngineer}$ = {*flexible, team oriented, positive attitude, systematic and organized, respects the others, seeking for knowledge, convincing ability, ability to interact with managers and customers*}

Having the personal competencies we had to make an effective analysis over the corresponding responsibilities and in this way to find the factors that influence at most the individuals. We have adopted the FMEA method for this goal as it gives the possibility to break each process into its small peaces and to look inside for possible failure modes and their causes. The analysis of the software team roles involved in the typical software engineering process ended with the discovery of the human factors that influence the different potential failure modes, which can be seen in the following list:

1. Coordination
2. Fear
3. Self-management
4. Management skills
5. Mental Overload=Stress
6. Intelligence
7. Competence
8. Analysis skills
9. Knowledge
10. Openness
11. Effectiveness
12. Creativity
13. Concentration
14. Emotional stability
15. Communication
16. Judgment
17. Self-Development
18. Problem solving ability
19. Liberalism
20. Perception
21. Control delegation
22. Professionalism
23. Selfish=Egoism
24. Persistence
25. Over self-confident
26. Dutifulness
27. Self-organization
28. Motivation
29. Hardworking
30. Achievement
31. Attention
32. Responsibility
33. Conscientiousness
34. Talkativeness
35. Leader skills
36. Personal attitude
37. Experience
38. Technical understanding
39. Personal grow
40. Imagination

41. Understanding ability
42. Patience
43. Planning skills
44. Friendliness
45. Observing ability
46. Cooperation
47. Appreciation

Having all the critical human factors for the software process we were faced with a new problem. How can we measure these traits and how can we examine a person in order to be able to understand which features does he posses and into which extent so that we can find out how they influence his work performance.

For this purpose we have adopted a well-accepted method in the Personality Evaluation – the Big Five theory. Of course we had to change it so that it could be applied in the software engineering field and after that we were able to define seven psychological characteristics that are complex enough to be matched with the discovered human factors and be used for the description of the personality features and the software productiveness. They are as follows:

1. Openness (Intelligence)
2. Conscientiousness
3. Extroversion
4. Agreeableness
5. Neuroticism (Emotional Stability)
6. Experience
7. Motivation

Analyzing the characteristics and the type of connection between them and the human performance in the IT, we have decided to design the whole process as an Experiment and to analyze it in order to model the desired dependence.

We have conducted three experiments by three special values of the motivation factor, because of the complex subjective-psychological dependence between motivation and performance. For these three experiments we have gained three statistically correct mathematical models, which describe the connection between the psychological characteristics (Motivation, Conscientiousness; Openness and Agreeableness) and the performance in the software engineering.

They are as follows:

*Performance by Motivation of 55%* = pr(co, int, agr) = -523.021607 + 3.139297*co + 7.793311*int + 4.525325*agr – 0.002677*co*int + 0.001674*co*agr – 0.021694*co*co – 0.046799*int*int – 0.031346*agr*agr

*Performance by Motivation of 70%* = pr(co, int, agr) = -611.111026 + 3.619927*co + 9.066844*int + 5.388229*agr – 0.001979*co*int + 0.001319*co*agr – 0.025236*co*co – 0.054906*int*int – 0.036977*agr*agr

*Performance by Motivation of 85%* = pr(co, int, agr) = -591.921937 + 3.671524*co + 8.791920*int + 5.377006*agr – 0.003125*co*int + 0.000694*co*agr+0.000833*int – 0.024559*co*co – 0.053047*int*int – 0.037094*agr*agr

The decision to make three experiments came from the observation that: by 55%; 70% and 85% of motivation can be seen a significant change in the performance values. The growing of the motivation up to 85% is connected with growing of the desire to give the best possible productiveness at work and by higher values than 85% this desire is decreasing. This can be explained with the fact that the people with 100% of motivation find it difficult to see perspective for development as they have already reached the maximum; this is a kind of de-motivation and results in lower performance levels.

The most important result from the models are the dependencies between the three examined features and the human performance:

- *The connection btw agreeableness and performance:* the growing values of agreeableness up to 75% characterize with growing of the performance because the employee is able to communicate and cooperate with his colleagues, he is able to accept others' ideas and to follow instructions; after these values the person loses his own judgment and cannot resolve any problem alone. The software engineer agrees with everyone and is not able to take decisions anymore and this leads to low values of performance. When the values are low around 40%, he is not able to cooperate and works very difficult with other people and this of course means also low productivity.

- *The connection btw intelligence and performance:* with the increasing of the intelligence after a specific point (around 80%), the observed software team members start to take very complex decisions and don't choose the optimal algorithm for resolving a problem. This leads to complications and more mistakes in the work process, the employees need more time and the solutions are not optimal, because of this it is logical to observe the decreasing of the performance. By low values, even by 60% we have very low productiveness which shows that we need employees with intelligence over the average in order to manage the software engineering process.

- *The connection btw conscientiousness and performance:* by growing until 70% it shows that the software specialist is trying to do his best and to manage his work as good as possible. From other side this characteristic hinders the process of ignoring the unimportant details in the everyday work, and exactly this leads to decreasing of the performance, when the conscientiousness is higher than 70%. The employee loses too much time in checking details and spending time for not so important problems which needs more time and results into lower productivity. When the values are low, until 55% we have very low performance and this is to be explained with the fact that such employees are not doing their job with the needed respect and cautious.

The results of the whole development process of the predictive model can be characterized in the following manner:

$$^{eval(BigFive)}_{DoE}HF^{FMEA}_{SoftwareProcess} = \mathcal{F}(Motivation,\ Conscientiousness,\ Openness,\ Agreeableness)$$

This IT human factors evaluation approach could be summarized in a simplified manner given in the following figure 65.

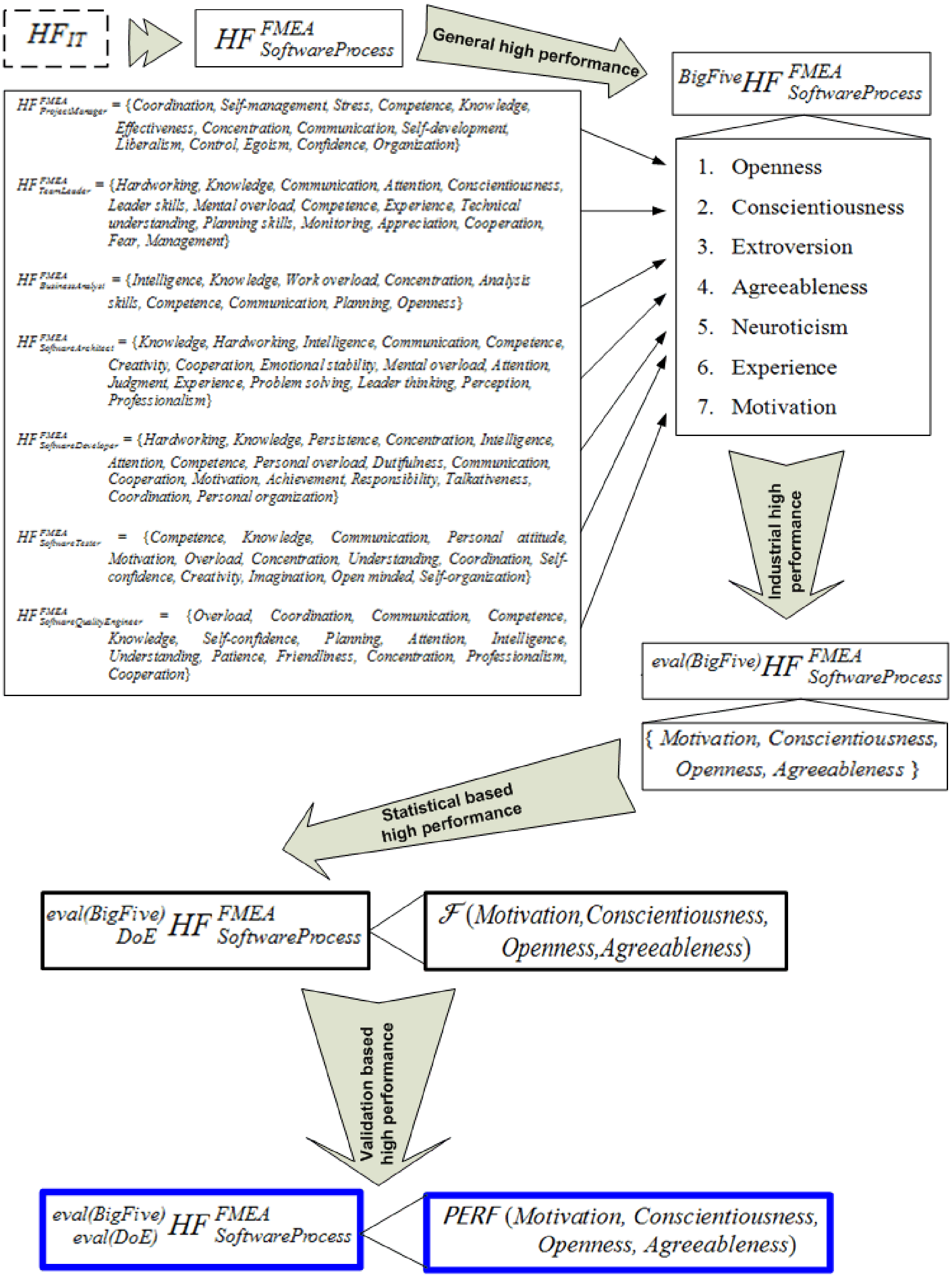



**Figure 65 The IT human factors approach keeping high performance**

The developed models were shown to be correct and adequate using special statistical formulas and further on with the development of special software. The web-application aims to show once again the models' correctness and effectiveness. The developed web-site represents a test with the special questions needed for the personnel evaluation and after this uses this data as an input for the developed model. In this way the model can be very easily used in the process of recruitment for selecting the best employees for a specific company.

Everything that we have said until the moment shows that:

- The problem described in the beginning of the PhD work: the connection between the personality and the individuals' performance in the software engineering is found!
- A method that models the human performance in the IT based on the specific psychological traits has been developed.
- We have gained three different models by the special values of Motivation that calculate the expected Performance.
- The developed model was tested and validated in real conditions and proved its correctness and usefulness for the software development.
- The model is an absolutely new scientific contribution that is extremely important for the process of improving the IT recruitment process.
- The developed method can be used also for prognosis of the productiveness of the whole software company based on the performance of the individuals.
- A modelling of the critical psychological features, which take part in the model, is another idea that can be applied and in this way can be increased the expected performance.

The scientific work in this PhD thesis makes the following ***main contributions*** to research within the field of software engineering:

1. An up-to-date review over the software risk assessment methods with special focus over their incompleteness.
2. An up-to-date review over the methods for employee evaluation and research over their existence and application in the software engineering.
3. Detailed research over the concepts in the software field and over the software team members with their specific capabilities and responsibilities.
4. Development of new scientific method 'Software Human Factor FMEA' for the extraction of the critical human factors.
5. Development of new scientific method for the evaluation of the human psychological features in the IT (with the adoption of the Big Five theory in the software engineering).
6. Development of unique mathematical model for the prediction of the individuals' performance in the IT based on his/her personal characteristics.
7. Development of web-based application realizing the mathematical model and supporting the software engineering research with concrete tool for the employee evaluation and personnel selection.

***Future work*** considering our approach in order to improve human factor involvements in the IT area will be:

- *Further application areas*: the specification of the method for each role in the software development process for different software process approaches (like agile development, V&V teams, PSP and collaborative software evolution).
- *Methodology improvements*: the actual application of this new approach considers a special kind of systems and software processes and should be extended by experience repositories like SLIM, ISBSG and QSM.
- *Team-oriented model extensions*: the current new approach supports the evaluation process of the IT personal and should be extended by further involvements of team characteristics (like pair programming, test teams and egoless approaches).
- *Human characteristics modelling*: the development model can be extended with additional methods for influencing the individuals' psychological traits. In this way the productiveness will be increased with stimulating the Motivation for example.
- *Whole evaluation:* a whole assessment of the software company can be build, based on the performance of each individual.

We can summarize once again with the following:

The developed mathematical model gives the possibility to predict the productiveness of the examined person based on his/her special psychological traits. This supports the process of IT personnel recruitment and also the whole process of IT personnel development with a powerful tool for achieving of better software quality. The right people chosen in the right manner and also their motivation are the most important software resources, crucial for the achievement of better results in the IT field.

# Appendix:

## One-dimensional intersections by Motivation of 55%

By Motivation of 55% we can observe the following one-dimensional intersections of the two-dimensional ones.

We can see here three one-dimensional intersections by all possible combinations with X1 = -1; X1 = 0 and X1 = +1.

| | | | |
|---|---|---|---|
| Построяване на едномерно сечение за X1 = 0, X2 = -1 | | | |
| int = 68.0000 [ c ] | Y = 61.6056 +3.6599 * X3 -7.052851 * X3*X3 | X3ек. = 0.2595 | agrек. = 73.8920 |
| | pr = -104.4371 +4.6324 * agr -0.0313 * agr * agr | Yек. = 62.0804 | prек. = 62.0804 |
| One dimensional intersection at X1 = 0, X2 = 0 | | | |
| int = 78.0000 [ c ] | Y = 69.498523 +3.659933 * X3 -7.052851 * X3*X3 | X3ек. = 0.2595 | agrек. = 73.8920 |
| | pr = -101.1766 +4.6324 * agr -0.0313 * agr * agr | Yек. = 69.9733 | prек. = 69.9733 |
| Построяване на едномерно сечение за X1 = 0, X2 = +1 | | | |
| int = 88.0000 [ b ] | Y = 68.0316 +3.6599 * X3 -7.052851 * X3*X3 | X3ек. = 0.2595 | agrек. = 73.8920 |
| | pr = -102.6435 +4.6324 * agr -0.0313 * agr * agr | Yек. = 68.5064 | prек. = 68.5064 |
| Построяване на едномерно сечение за X1 = 0, X3 = -1 | | | |
| agr = 55.0000 [ t ] | Y = 58.7857 +3.2130 * X2 -4.679923 * X2*X2 | X2ек. = 0.3433 | intек. = 81.4327 |
| | pr = -251.0020 +7.6220 * int -0.046799 * int * int | Yек. = 59.3372 | prек. = 59.3372 |
| Построяване на едномерно сечение за X1 = 0, X3 = 0 | | | |
| agr = 70.0000 [ t ] | Y = 69.498523 +3.212974 * X2 -4.679923 * X2*X2 | X2ек. = 0.3433 | intек. = 81.4327 |
| | pr = -240.2892 +7.6220 * int -0.046799 * int * int | Yек. = 70.0500 | prек. = 70.0500 |
| Построяване на едномерно сечение за X1 = 0, X3 = +1 | | | |
| agr = 85.0000 [ t ] | Y = 66.1056 +3.6599 * X2 -4.679923 * X2*X2 | X2ек. = 0.3910 | intек. = 81.9102 |
| | pr = -247.1684 +7.6667 * int -0.046799 * int * int | Yек. = 66.8212 | prек. = 66.8212 |

| | | | |
|---|---|---|---|
| Едномерно сечение при X1 = -1, X2 = -1 | | | |
| int = 68.0000 [ c ] | Y = 55.2317 +3.6599 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2595 | agr(екс.) = 73.8920 |
| | pr = -110.8110 +4.6324 * agr -0.0313 * agr * agr | Y(екс.) = 55.7065 | pr(екс.) = 55.7065 |
| Едномерно сечение при X1 = -1, X2 = 0 | | | |
| int = 78.0000 [ c ] | Y = 69.498523 +3.659933 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2595 | agr(екс.) = 73.8920 |
| | pr = -107.5505 +4.6324 * agr -0.0313 * agr * agr | Y(екс.) = 63.5994 | pr(екс.) = 63.5994 |
| Едномерно сечение при X1 = -1, X2 = +1 | | | |
| int = 88.0000 [ b ] | Y = 61.6577 +3.6599 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2595 | agr(екс.) = 73.8920 |
| | pr = -109.0174 +4.6324 * agr -0.0313 * agr * agr | Y(екс.) = 62.1325 | pr(екс.) = 62.1325 |
| Едномерно сечение при X1 = -1, X3 = -1 | | | |
| agr = 55.0000 [ t ] | Y = 52.4118 +3.2130 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3433 | int(екс.) = 81.4327 |
| | pr = -257.3759 +7.6220 * int -0.046799 * int * int | Y(екс.) = 52.9633 | pr(екс.) = 52.9633 |
| Едномерно сечение при X1 = -1, X3 = 0 | | | |
| agr = 70.0000 [ t ] | Y = 69.498523 +3.212974 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3433 | int(екс.) = 81.4327 |
| | pr = -246.6631 +7.6220 * int -0.046799 * int * int | Y(екс.) = 63.6761 | pr(екс.) = 63.6761 |
| Едномерно сечение при X1 = -1, X3 = +1 | | | |
| agr = 85.0000 [ t ] | Y = 59.7317 +3.6599 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3910 | int(екс.) = 81.9102 |
| | pr = -253.5423 +7.6667 * int -0.046799 * int * int | Y(екс.) = 60.4473 | pr(екс.) = 60.4473 |

| Сечение | Фиксиран фактор | Уравнения | Екстремум | Екстремум |
|---|---|---|---|---|
| Едномерно сечение при X1 = +1, X2 = -1 | int = 68.0000 [ c ] | Y = 61.7317 +3.6599 * X3 -7.052851 * X3*X3<br>pr = -104.3110 +4.6324 * agr -0.0313 * agr * agr | X3(екс.) = 0.2595<br>Y(екс.) = 62.2065 | agr(екс.) = 73.8920<br>pr(екс.) = 62.2065 |
| Едномерно сечение при X1 = +1, X2 = 0 | int = 78.0000 [ c ] | Y = 69.498523 +3.659933 * X3 -7.052851 * X3*X3<br>pr = -101.0505 +4.6324 * agr -0.0313 * agr * agr | X3(екс.) = 0.2595<br>Y(екс.) = 70.0994 | agr(екс.) = 73.8920<br>pr(екс.) = 70.0994 |
| Едномерно сечение при X1 = +1, X2 = +1 | int = 88.0000 [ b ] | Y = 68.1577 +3.6599 * X3 -7.052851 * X3*X3<br>pr = -102.5174 +4.6324 * agr -0.0313 * agr * agr | X3(екс.) = 0.2595<br>Y(екс.) = 68.6325 | agr(екс.) = 73.8920<br>pr(екс.) = 68.6325 |
| Едномерно сечение при X1 = +1, X3 = -1 | agr = 55.0000 [ t ] | Y = 58.9118 +3.2130 * X2 -4.679923 * X2*X2<br>pr = -250.8759 +7.6220 * int -0.046799 * int * int | X2(екс.) = 0.3433<br>Y(екс.) = 59.4633 | int(екс.) = 81.4327<br>pr(екс.) = 59.4633 |
| Едномерно сечение при X1 = +1, X3 = 0 | agr = 70.0000 [ t ] | Y = 69.498523 +3.212974 * X2 -4.679923 * X2*X2<br>pr = -240.1631 +7.6220 * int -0.046799 * int * int | X2(екс.) = 0.3433<br>Y(екс.) = 70.1761 | int(екс.) = 81.4327<br>pr(екс.) = 70.1761 |
| Едномерно сечение при X1 = +1, X3 = +1 | agr = 85.0000 [ t ] | Y = 66.2317 +3.6599 * X2 -4.679923 * X2*X2<br>pr = -247.0423 +7.6667 * int -0.046799 * int * int | X2(екс.) = 0.3910<br>Y(екс.) = 66.9473 | int(екс.) = 81.9102<br>pr(екс.) = 66.9473 |

We can see here three one-dimensional intersections by all possible combinations with X2 = -1; X2 = 0 and X2 = +1.

| Сечение | Фиксиран фактор | Уравнения | Екстремум | Екстремум |
|---|---|---|---|---|
| Едномерно сечение при X2 = -1, X1 = -1 | co = 52.0000 [ pr ] | Y = 53.7127 +3.3587 * X3 -7.052851 * X3*X3<br>pr = -109.4252 +4.6124 * agr -0.0313 * agr * agr | X3(екс.) = 0.2381<br>Y(екс.) = 54.1126 | agr(екс.) = 73.5716<br>pr(екс.) = 54.1126 |
| Едномерно сечение при X2 = -1, X1 = 0 | co = 64.0000 [ pr ] | Y = 69.498523 +3.659933 * X3 -7.052851 * X3*X3<br>pr = -109.0695 +4.6324 * agr -0.0313 * agr * agr | X3(екс.) = 0.2595<br>Y(екс.) = 62.0804 | agr(екс.) = 73.8920<br>pr(екс.) = 62.0804 |
| Едномерно сечение при X2 = -1, X1 = +1 | co = 76.0000 [ pr ] | Y = 60.1387 +3.9612 * X3 -7.052851 * X3*X3<br>pr = -111.9423 +4.6525 * agr -0.0313 * agr * agr | X3(екс.) = 0.2808<br>Y(екс.) = 60.6949 | agr(екс.) = 74.2123<br>pr(екс.) = 60.6949 |
| Едномерно сечение при X2 = -1, X3 = -1 | agr = 55.0000 [ t ] | Y = 50.8928 +2.9487 * X1 -3.123905 * X1*X1<br>pr = -53.6916 +3.0225 * co -0.021694 * co * co | X1(екс.) = 0.4720<br>Y(екс.) = 51.5887 | co(екс.) = 69.6636<br>pr(екс.) = 51.5887 |
| Едномерно сечение при X2 = -1, X3 = 0 | agr = 70.0000 [ t ] | Y = 69.498523 +3.249996 * X1 -3.123905 * X1*X1<br>pr = -44.5854 +3.0476 * co -0.021694 * co * co | X1(екс.) = 0.5202<br>Y(екс.) = 62.4509 | co(екс.) = 70.2422<br>pr(екс.) = 62.4509 |
| Едномерно сечение при X2 = -1, X3 = +1 | agr = 85.0000 [ t ] | Y = 58.2127 +3.5512 * X1 -3.123905 * X1*X1<br>pr = -49.5850 +3.0727 * co -0.021694 * co * co | X1(екс.) = 0.5684<br>Y(екс.) = 59.2220 | co(екс.) = 70.8208<br>pr(екс.) = 59.2220 |

Едномерно сечение при X2 = 0, X1 = -1

co = 54.0000 [ b ]

Y = 63.1246 +3.3587 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2381 | agr(екс.) = 73.5716

pr = -101.5323 +4.6124 * agr -0.0313 * agr * agr | Y(екс.) = 63.5245 | pr(екс.) = 63.5245

Едномерно сечение при X2 = 0, X1 = 0

co = 64.0000 [ b ]

Y = 69.498523 +3.659933 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2595 | agr(екс.) = 73.8920

pr = -101.1766 +4.6324 * agr -0.0313 * agr * agr | Y(екс.) = 69.9733 | pr(екс.) = 69.9733

Едномерно сечение при X2 = 0, X1 = +1

co = 74.0000 [ b ]

Y = 69.6246 +3.9612 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2808 | agr(екс.) = 74.2123

pr = -102.4563 +4.6525 * agr -0.0313 * agr * agr | Y(екс.) = 70.1808 | pr(екс.) = 70.1808

Едномерно сечение при X2 = 0, X3 = -1

agr = 55.0000 [ t ]

Y = 58.7857 +2.9487 * X1 -3.123905 * X1*X1 | X2(екс.) = 0.4720 | co(екс.) = 69.5925

pr = -45.7987 +3.0225 * co -0.021694 * co * co | Y(екс.) = 59.4816 | pr(екс.) = 59.4816

Едномерно сечение при X2 = 0, X3 = 0

agr = 70.0000 [ t ]

Y = 69.498523 +3.249996 * X1 -3.123905 * X1*X1 | X2(екс.) = 0.5202 | co(екс.) = 70.2422

pr = -80.7552 +4.4277 * co -0.032499 * co * co | Y(екс.) = 70.3438 | pr(екс.) = 70.2460

Едномерно сечение при X2 = 0, X3 = +1

agr = 85.0000 [ t ]

Y = 66.1056 +3.5512 * X1 -3.123905 * X1*X1 | X2(екс.) = 0.5684 | co(екс.) = 70.8208

pr = -41.6921 +3.0727 * co -0.021694 * co * co | Y(екс.) = 67.1149 | pr(екс.) = 67.1149

Едномерно сечение при X2 = +1, X1 = -1

co = 52.0000 [ b ]

Y = 60.1387 +3.3587 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2381 | agr(екс.) = 73.5716

pr = -104.5183 +4.6124 * agr -0.0313 * agr * agr | Y(екс.) = 60.5385 | pr(екс.) = 60.5385

Едномерно сечение при X2 = +1, X1 = 0

co = 64.0000 [ b ]

Y = 69.498523 +3.659933 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2595 | agr(екс.) = 73.8920

pr = -102.6435 +4.6324 * agr -0.0313 * agr * agr | Y(екс.) = 68.5064 | pr(екс.) = 68.5064

Едномерно сечение при X2 = +1, X1 = +1

co = 76.0000 [ b ]

Y = 66.5646 +3.9612 * X3 -7.052851 * X3*X3 | X3(екс.) = 0.2808 | agr(екс.) = 74.2123

pr = -105.5163 +4.6525 * agr -0.0313 * agr * agr | Y(екс.) = 67.1208 | pr(екс.) = 67.1208

Едномерно сечение при X2 = +1, X3 = -1

agr = 55.0000 [ t ]

Y = 57.3188 +2.9487 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.4720 | co(екс.) = 69.6636

pr = -47.2656 +3.0225 * int -0.021694 * int * int | Y(екс.) = 58.0146 | pr(екс.) = 58.0146

Едномерно сечение при X2 = +1, X3 = 0

agr = 70.0000 [ t ]

Y = 69.498523 +3.249996 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.5202 | co(екс.) = 70.2422

pr = -38.1595 +3.0476 * int -0.021694 * int * int | Y(екс.) = 68.8769 | pr(екс.) = 68.8769

Едномерно сечение при X2 = +1, X3 = +1

agr = 85.0000 [ t ]

Y = 64.6387 +3.5512 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.5684 | co(екс.) = 70.8208

pr = -43.1591 +3.0727 * int -0.021694 * int * int | Y(екс.) = 65.6479 | pr(екс.) = 65.6479

We can see here three one-dimensional intersections by all possible combinations with X3 = -1; X3 = 0 and X3 = +1.

| | | | |
|---|---|---|---|
| Едномерно сечение при X3 = -1, X1 = -1 | | | |
| co = 52.0000 [ pr ] | Y = 52.4118 +3.5342 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3776 | int(екс.) = 81.7759 |
| | pr = -252.2275 +7.6541 * int -0.0468 * int * int | Y(екс.) = 53.0791 | pr(екс.) = 53.0791 |
| Едномерно сечение при X3 = -1, X1 = 0 | | | |
| co = 64.0000 [ pr ] | Y = 69.498523 +3.212974 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3433 | int(екс.) = 81.4327 |
| | pr = -251.0020 +7.6220 * int -0.0468 * int * int | Y(екс.) = 59.3372 | pr(екс.) = 59.3372 |
| Едномерно сечение при X3 = -1, X1 = +1 | | | |
| co = 76.0000 [ pr ] | Y = 58.9118 +2.8917 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3089 | int(екс.) = 81.0895 |
| | pr = -248.3702 +7.5899 * int -0.0468 * int * int | Y(екс.) = 59.3585 | pr(екс.) = 59.3585 |
| Едномерно сечение при X3 = -1, X2 = -1 | | | |
| int = 68.0000 [ c ] | Y = 50.8928 +3.5712 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.5716 | co(екс.) = 70.8592 |
| | pr = -57.0116 +3.0744 * co -0.021694 * co * co | Y(екс.) = 51.9135 | pr(екс.) = 51.9135 |
| Едномерно сечение при X3 = -1, X2 = 0 | | | |
| int = 78.0000 [ c ] | Y = 69.498523 +3.249996 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.5202 | co(екс.) = 70.2422 |
| | pr = -47.4053 +3.0476 * co -0.021694 * co * co | Y(екс.) = 59.6310 | pr(екс.) = 59.6310 |
| Едномерно сечение при X3 = -1, X2 = +1 | | | |
| int = 88.0000 [ c ] | Y = 57.3188 +2.9287 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.4688 | co(екс.) = 69.6252 |
| | pr = -47.1589 +3.0209 * co -0.021694 * co * co | Y(екс.) = 58.0052 | pr(екс.) = 58.0052 |

| | | | |
|---|---|---|---|
| Едномерно сечение при X3 = 0, X1 = -1 | | | |
| co = 52.0000 [ b ] | Y = 63.1246 +3.5342 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3776 | int(екс.) = 81.7759 |
| | pr = -241.5148 +7.6541 * int -0.0468 * int * int | Y(екс.) = 63.7919 | pr(екс.) = 63.7919 |
| Едномерно сечение при X3 = 0, X1 = 0 | | | |
| co = 64.0000 [ b ] | Y = 69.498523 +3.212974 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3433 | int(екс.) = 81.4327 |
| | pr = -240.2892 +7.6220 * int -0.0468 * int * int | Y(екс.) = 70.0500 | pr(екс.) = 70.0500 |
| Едномерно сечение при X3 = 0, X1 = +1 | | | |
| co = 76.0000 [ b ] | Y = 69.6246 +2.8917 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3089 | int(екс.) = 81.0895 |
| | pr = -237.6574 +7.5899 * int -0.0468 * int * int | Y(екс.) = 70.0713 | pr(екс.) = 70.0713 |
| Едномерно сечение при X3 = 0, X2 = -1 | | | |
| int = 68.0000 [ c ] | Y = 61.6056 +3.5712 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.5716 | co(екс.) = 70.7881 |
| | pr = -46.2988 +3.0744 * co -0.021694 * co * co | Y(екс.) = 62.6263 | pr(екс.) = 62.6263 |
| Едномерно сечение при X3 = 0, X2 = 0 | | | |
| int = 78.0000 [ t ] | Y = 69.498523 +3.249996 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.5202 | co(екс.) = 70.2422 |
| | pr = -36.6925 +3.0476 * co -0.021694 * co * co | Y(екс.) = 70.3438 | pr(екс.) = 70.3438 |
| Едномерно сечение при X3 = 0, X2 = +1 | | | |
| int = 88.0000 [ t ] | Y = 68.0316 +2.9287 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.4688 | co(екс.) = 69.6252 |
| | pr = -36.4462 +3.0209 * co -0.021694 * co * co | Y(екс.) = 68.7180 | pr(екс.) = 68.7180 |

| | | | |
|---|---|---|---|
| **Едномерно сечение при X3 = +1, X1 = -1** | | | |
| co = 52.0000 [ b ] | Y = 59.7317 +3.5342 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3776 | int(екс.) = 81.7759 |
| | pr = -244.9077 +7.6541 * int -0.0468 * int * int | Y(екс.) = 60.3990 | pr(екс.) = 60.3990 |
| **Едномерно сечение при X3 = +1, X1 = 0** | | | |
| co = 64.0000 [ b ] | Y = 69.498523 +3.212974 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3433 | int(екс.) = 81.4327 |
| | pr = -243.6821 +7.6220 * int -0.0468 * int * int | Y(екс.) = 66.6571 | pr(екс.) = 66.6571 |
| **Едномерно сечение при X3 = +1, X1 = +1** | | | |
| co = 76.0000 [ b ] | Y = 66.2317 +2.8917 * X2 -4.679923 * X2*X2 | X2(екс.) = 0.3089 | int(екс.) = 81.0895 |
| | pr = -241.0503 +7.5899 * int -0.0468 * int * int | Y(екс.) = 66.6784 | pr(екс.) = 66.6784 |
| **Едномерно сечение при X3 = +1, X2 = -1** | | | |
| int = 66.0000 [ c ] | Y = 55.3928 +3.5712 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.5716 | co(екс.) = 70.8592 |
| | pr = -49.6917 +3.0744 * co -0.021694 * co * co | Y(екс.) = 59.2334 | pr(екс.) = 59.2334 |
| **Едномерно сечение при X3 = +1, X2 = 0** | | | |
| int = 78.0000 [ c ] | Y = 69.498523 +3.249996 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.5202 | co(екс.) = 70.2422 |
| | pr = -40.0855 +3.0476 * co -0.021694 * co * co | Y(екс.) = 66.9509 | pr(екс.) = 66.9509 |
| **Едномерно сечение при X3 = +1, X2 = +1** | | | |
| int = 88.0000 [ c ] | Y = 64.6387 +2.9287 * X1 -3.123905 * X1*X1 | X1(екс.) = 0.4688 | co(екс.) = 69.6252 |
| | pr = -39.8391 +3.0209 * co -0.021694 * co * co | Y(екс.) = 65.3251 | pr(екс.) = 65.3251 |

**One-dimensional intersections by Motivation of 70%**

<u>By Motivation of 70% we can observe the following one-dimensional intersections of the two-dimensional ones.</u>

We can see here three one-dimensional intersections by all possible combinations with X1 = -1; X1 = 0 and X1 = +1.

**Едномерно сечение при X1 = -1, X2 = -1**

int = 68.0000 [ c ]

Y = 65.5789 +4.4392 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018

pr = -130.8502 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 66.1711 | pr(екс.) = 66.1711

**Едномерно сечение при X1 = -1, X2 = 0**

int = 78.0000 [ c ]

Y = 82.384068 +4.439234 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018

pr = -127.0840 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 75.4099 | pr(екс.) = 75.4099

**Едномерно сечение при X1 = -1, X2 = +1**

int = 88.0000 [ b ]

Y = 73.0754 +4.4392 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018

pr = -128.8264 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 73.6676 | pr(екс.) = 73.6676

**Едномерно сечение при X1 = -1, X3 = -1**

agr = 55.0000 [ t ]

Y = 62.0588 +3.7482 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3413 | int(екс.) = 81.4133

pr = -301.2263 +8.9402 * int -0.054906 * int * int | Y(екс.) = 62.6985 | pr(екс.) = 62.6985

**Едномерно сечение при X1 = -1, X3 = 0**

agr = 70.0000 [ t ]

Y = 82.384068 +3.748234 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3413 | int(екс.) = 81.4133

pr = -288.4673 +8.9402 * int -0.054906 * int * int | Y(екс.) = 75.4575 | pr(екс.) = 75.4575

**Едномерно сечение при X1 = -1, X3 = +1**

agr = 85.0000 [ t ]

Y = 70.9373 +4.4392 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.4043 | int(екс.) = 82.0426

pr = -297.7376 +9.0093 * int -0.054906 * int * int | Y(екс.) = 71.8346 | pr(екс.) = 71.8346

---

**Построяване на едномерно сечение за X1 = 0, X2 = -1**

int = 68.0000 [ c ]

Y = 73.1452 +4.4392 * X3 -8.319736 * X3*X3 | X3ек. = 0.2668 | agrек. = 74.0018

pr = -123.2839 +5.4727 * agr -0.0370 * agr * agr | Yек. = 73.7374 | prек. = 73.7374

**One dimensional intersection at X1 = 0, X2 = 0**

int = 78.0000 [ c ]

Y = 82.384068 +4.439234 * X3 -8.319736 * X3*X3 | X3ек. = 0.2668 | agrек. = 74.0018

pr = -119.5177 +5.4727 * agr -0.0370 * agr * agr | Yек. = 82.9762 | prек. = 82.9762

**Построяване на едномерно сечение за X1 = 0, X2 = +1**

int = 88.0000 [ b ]

Y = 80.6417 +4.4392 * X3 -8.319736 * X3*X3 | X3ек. = 0.2668 | agrек. = 74.0018

pr = -121.2601 +5.4727 * agr -0.0370 * agr * agr | Yек. = 81.2339 | prек. = 81.2339

**Построяване на едномерно сечение за X1 = 0, X3 = -1**

agr = 55.0000 [ t ]

Y = 69.6251 +3.7482 * X2 -5.490612 * X2*X2 | X2ек. = 0.3413 | intек. = 81.4133

pr = -293.6599 +8.9402 * int -0.054906 * int * int | Yек. = 70.2648 | prек. = 70.2648

**Построяване на едномерно сечение за X1 = 0, X3 = 0**

agr = 70.0000 [ t ]

Y = 82.384068 +3.748234 * X2 -5.490612 * X2*X2 | X2ек. = 0.3413 | intек. = 81.4133

pr = -280.9010 +8.9402 * int -0.054906 * int * int | Yек. = 83.0238 | prек. = 83.0238

**Построяване на едномерно сечение за X1 = 0, X3 = +1**

agr = 85.0000 [ t ]

Y = 78.5036 +4.4392 * X2 -5.490612 * X2*X2 | X2ек. = 0.4043 | intек. = 82.0426

pr = -290.1713 +9.0093 * int -0.054906 * int * int | Yек. = 79.4009 | prек. = 79.4009

| | | | |
|---|---|---|---|
| **Едномерно сечение при X1 = +1, X2 = -1** | | | |
| int = 68.0000 [ c ] | Y = 73.4435 +4.4392 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018 |
| | pr = -122.9856 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 74.0357 | pr(екс.) = 74.0357 |
| **Едномерно сечение при X1 = +1, X2 = 0** | | | |
| int = 78.0000 [ c ] | Y = 82.384068 +4.439234 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018 |
| | pr = -119.2194 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 83.2745 | pr(екс.) = 83.2745 |
| **Едномерно сечение при X1 = +1, X2 = +1** | | | |
| int = 88.0000 [ b ] | Y = 80.9400 +4.4392 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018 |
| | pr = -120.9618 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 81.5322 | pr(екс.) = 81.5322 |
| **Едномерно сечение при X1 = +1, X3 = -1** | | | |
| agr = 55.0000 [ t ] | Y = 69.9234 +3.7482 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3413 | int(екс.) = 81.4133 |
| | pr = -293.3616 +8.9402 * int -0.054906 * int * int | Y(екс.) = 70.5631 | pr(екс.) = 70.5631 |
| **Едномерно сечение при X1 = +1, X3 = 0** | | | |
| agr = 70.0000 [ t ] | Y = 82.384068 +3.748234 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3413 | int(екс.) = 81.4133 |
| | pr = -280.6027 +8.9402 * int -0.054906 * int * int | Y(екс.) = 83.3221 | pr(екс.) = 83.3221 |
| **Едномерно сечение при X1 = +1, X3 = +1** | | | |
| agr = 85.0000 [ t ] | Y = 78.8019 +4.4392 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.4043 | int(екс.) = 82.0426 |
| | pr = -289.8730 +9.0093 * int -0.054906 * int * int | Y(екс.) = 79.6992 | pr(екс.) = 79.6992 |

We can see here three one-dimensional intersections by all possible combinations with X2 = -1; X2 = 0 and X2 = +1.

| | | | |
|---|---|---|---|
| **Едномерно сечение при X2 = -1, X1 = -1** | | | |
| co = 52.0000 [ pr ] | Y = 63.9064 +4.2017 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2525 | agr(екс.) = 73.7877 |
| | pr = -129.7577 +5.4568 * agr -0.0370 * agr * agr | Y(екс.) = 64.4369 | pr(екс.) = 64.4369 |
| **Едномерно сечение при X2 = -1, X1 = 0** | | | |
| co = 64.0000 [ pr ] | Y = 82.384068 +4.439234 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018 |
| | pr = -128.7566 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 73.7374 | pr(екс.) = 73.7374 |
| **Едномерно сечение при X2 = -1, X1 = +1** | | | |
| co = 76.0000 [ pr ] | Y = 71.4028 +4.6767 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2811 | agr(екс.) = 74.2159 |
| | pr = -131.6073 +5.4885 * agr -0.0370 * agr * agr | Y(екс.) = 72.0601 | pr(екс.) = 72.0601 |
| **Едномерно сечение при X2 = -1, X3 = -1** | | | |
| agr = 55.0000 [ t ] | Y = 60.3863 +3.6948 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5084 | co(екс.) = 70.1004 |
| | pr = -62.6865 +3.5381 * co -0.025236 * co * co | Y(екс.) = 61.3254 | pr(екс.) = 61.3254 |
| **Едномерно сечение при X2 = -1, X3 = 0** | | | |
| agr = 70.0000 [ t ] | Y = 82.384068 +3.932305 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5410 | co(екс.) = 70.4925 |
| | pr = -51.1942 +3.5579 * co -0.025236 * co * co | Y(екс.) = 74.2090 | pr(екс.) = 74.2090 |
| **Едномерно сечение при X2 = -1, X3 = +1** | | | |
| agr = 85.0000 [ t ] | Y = 69.2647 +4.1698 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5737 | co(екс.) = 70.8847 |
| | pr = -56.3413 +3.5777 * co -0.025236 * co * co | Y(екс.) = 70.4609 | pr(екс.) = 70.4609 |

Едномерно сечение при X2 = 0, X1 = -1

co = 54.0000 [ b ]

Y = 74.8178 +4.2017 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2525 | agr(екс.) = 73.7877

pr = -120.5188 +5.4568 * agr -0.0370 * agr * agr | Y(екс.) = 75.3483 | pr(екс.) = 75.3483

Едномерно сечение при X2 = 0, X1 = 0

co = 64.0000 [ b ]

Y = 82.384068 +4.439234 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018

pr = -119.5177 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 82.9762 | pr(екс.) = 82.9762

Едномерно сечение при X2 = 0, X1 = +1

co = 74.0000 [ b ]

Y = 82.6824 +4.6767 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2811 | agr(екс.) = 74.2159

pr = -120.3277 +5.4885 * agr -0.0370 * agr * agr | Y(екс.) = 83.3396 | pr(екс.) = 83.3396

Едномерно сечение при X2 = 0, X3 = -1

agr = 55.0000 [ t ]

Y = 69.6251 +3.6948 * X1 -3.633999 * X1*X1 | X2(екс.) = 0.5084 | co(екс.) = 69.7965

pr = -53.4476 +3.5381 * co -0.025236 * co * co | Y(екс.) = 70.5643 | pr(екс.) = 70.5643

Едномерно сечение при X2 = 0, X3 = 0

agr = 70.0000 [ t ]

Y = 82.384068 +3.932305 * X1 -3.633999 * X1*X1 | X2(екс.) = 0.5410 | co(екс.) = 70.4925

pr = -93.7839 +5.1929 * co -0.038129 * co * co | Y(екс.) = 83.4478 | pr(екс.) = 83.3029

Едномерно сечение при X2 = 0, X3 = +1

agr = 85.0000 [ t ]

Y = 78.5036 +4.1698 * X1 -3.633999 * X1*X1 | X2(екс.) = 0.5737 | co(екс.) = 70.8847

pr = -47.1025 +3.5777 * co -0.025236 * co * co | Y(екс.) = 79.6997 | pr(екс.) = 79.6997

Едномерно сечение при X2 = +1, X1 = -1

co = 52.0000 [ b ]

Y = 71.4028 +4.2017 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2525 | agr(екс.) = 73.7877

pr = -123.9338 +5.4568 * agr -0.0370 * agr * agr | Y(екс.) = 71.9333 | pr(екс.) = 71.9333

Едномерно сечение при X2 = +1, X1 = 0

co = 64.0000 [ b ]

Y = 82.384068 +4.439234 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2668 | agr(екс.) = 74.0018

pr = -121.2601 +5.4727 * agr -0.0370 * agr * agr | Y(екс.) = 81.2339 | pr(екс.) = 81.2339

Едномерно сечение при X2 = +1, X1 = +1

co = 76.0000 [ b ]

Y = 78.8993 +4.6767 * X3 -8.319736 * X3*X3 | X3(екс.) = 0.2811 | agr(екс.) = 74.2159

pr = -124.1108 +5.4885 * agr -0.0370 * agr * agr | Y(екс.) = 79.5565 | pr(екс.) = 79.5565

Едномерно сечение при X2 = +1, X3 = -1

agr = 55.0000 [ t ]

Y = 67.8827 +3.6948 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5084 | co(екс.) = 70.1004

pr = -55.1900 +3.5381 * int -0.025236 * int * int | Y(екс.) = 68.8219 | pr(екс.) = 68.8219

Едномерно сечение при X2 = +1, X3 = 0

agr = 70.0000 [ t ]

Y = 82.384068 +3.932305 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5410 | co(екс.) = 70.4925

pr = -43.6977 +3.5579 * int -0.025236 * int * int | Y(екс.) = 81.7055 | pr(екс.) = 81.7055

Едномерно сечение при X2 = +1, X3 = +1

agr = 85.0000 [ t ]

Y = 76.7612 +4.1698 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5737 | co(екс.) = 70.8847

pr = -48.8449 +3.5777 * int -0.025236 * int * int | Y(екс.) = 77.9573 | pr(екс.) = 77.9573

We can see here three one-dimensional intersections by all possible combinations with X3 = -1; X3 = 0 and X3 = +1.

Едномерно сечение при X3 = -1, X1 = -1
co = 52.0000 [ pr ]
Y = 62.0588 +3.9857 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3630 | int(екс.) = 81.6296
pr = -294.1148 +8.9639 * int -0.0549 * int * int | Y(екс.) = 62.7821 | pr(екс.) = 62.7821

Едномерно сечение при X3 = -1, X1 = 0
co = 64.0000 [ pr ]
Y = 82.384068 +3.748234 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3413 | int(екс.) = 81.4133
pr = -293.6599 +8.9402 * int -0.0549 * int * int | Y(екс.) = 70.2648 | pr(екс.) = 70.2648

Едномерно сечение при X3 = -1, X1 = +1
co = 76.0000 [ pr ]
Y = 69.9234 +3.5107 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3197 | int(екс.) = 81.1970
pr = -291.5091 +8.9164 * int -0.0549 * int * int | Y(екс.) = 70.4846 | pr(екс.) = 70.4846

Едномерно сечение при X3 = -1, X2 = -1
int = 68.0000 [ c ]
Y = 60.3863 +4.1698 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5737 | co(екс.) = 70.8847
pr = -65.2198 +3.5777 * co -0.025236 * co * co | Y(екс.) = 61.5824 | pr(екс.) = 61.5824

Едномерно сечение при X3 = -1, X2 = 0
int = 78.0000 [ c ]
Y = 82.384068 +3.932305 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5410 | co(екс.) = 70.4925
pr = -54.7143 +3.5579 * co -0.025236 * co * co | Y(екс.) = 70.6889 | pr(екс.) = 70.6889

Едномерно сечение при X3 = -1, X2 = +1
int = 88.0000 [ c ]
Y = 67.8827 +3.6948 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5084 | co(екс.) = 70.1004
pr = -55.1900 +3.5381 * co -0.025236 * co * co | Y(екс.) = 68.8219 | pr(екс.) = 68.8219

Едномерно сечение при X3 = 0, X1 = -1
co = 52.0000 [ b ]
Y = 74.8178 +3.9857 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3630 | int(екс.) = 81.6296
pr = -281.3559 +8.9639 * int -0.0549 * int * int | Y(екс.) = 75.5411 | pr(екс.) = 75.5411

Едномерно сечение при X3 = 0, X1 = 0
co = 64.0000 [ b ]
Y = 82.384068 +3.748234 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3413 | int(екс.) = 81.4133
pr = -280.9010 +8.9402 * int -0.0549 * int * int | Y(екс.) = 83.0238 | pr(екс.) = 83.0238

Едномерно сечение при X3 = 0, X1 = +1
co = 76.0000 [ b ]
Y = 82.6824 +3.5107 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3197 | int(екс.) = 81.1970
pr = -278.7502 +8.9164 * int -0.0549 * int * int | Y(екс.) = 83.2436 | pr(екс.) = 83.2436

Едномерно сечение при X3 = 0, X2 = -1
int = 68.0000 [ c ]
Y = 73.1452 +4.1698 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5737 | co(екс.) = 70.5807
pr = -52.4608 +3.5777 * co -0.025236 * co * co | Y(екс.) = 74.3414 | pr(екс.) = 74.3414

Едномерно сечение при X3 = 0, X2 = 0
int = 78.0000 [ t ]
Y = 82.384068 +3.932305 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5410 | co(екс.) = 70.4925
pr = -41.9553 +3.5579 * co -0.025236 * co * co | Y(екс.) = 83.4478 | pr(екс.) = 83.4478

Едномерно сечение при X3 = 0, X2 = +1
int = 88.0000 [ t ]
Y = 80.6417 +3.6948 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5084 | co(екс.) = 70.1004
pr = -42.4310 +3.5381 * co -0.025236 * co * co | Y(екс.) = 81.5808 | pr(екс.) = 81.5808

| | | | |
|---|---|---|---|
| Едномерно сечение при X3 = +1, X1 = -1 | | | |
| co = 52.0000 [ b ] | Y = 70.9373 +3.9857 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3630 | int(екс.) = 81.6296 |
| | pr = -285.2364 +8.9639 * int -0.0549 * int * int | Y(екс.) = 71.6606 | pr(екс.) = 71.6606 |
| Едномерно сечение при X3 = +1, X1 = 0 | | | |
| co = 64.0000 [ b ] | Y = 82.384068 +3.748234 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3413 | int(екс.) = 81.4133 |
| | pr = -284.7815 +8.9402 * int -0.0549 * int * int | Y(екс.) = 79.1433 | pr(екс.) = 79.1433 |
| Едномерно сечение при X3 = +1, X1 = +1 | | | |
| co = 76.0000 [ b ] | Y = 78.8019 +3.5107 * X2 -5.490612 * X2*X2 | X2(екс.) = 0.3197 | int(екс.) = 81.1970 |
| | pr = -282.6307 +8.9164 * int -0.0549 * int * int | Y(екс.) = 79.3631 | pr(екс.) = 79.3631 |
| Едномерно сечение при X3 = +1, X2 = -1 | | | |
| int = 66.0000 [ c ] | Y = 65.7446 +4.1698 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5737 | co(екс.) = 70.8847 |
| | pr = -56.3413 +3.5777 * co -0.025236 * co * co | Y(екс.) = 70.4609 | pr(екс.) = 70.4609 |
| Едномерно сечение при X3 = +1, X2 = 0 | | | |
| int = 78.0000 [ c ] | Y = 82.384068 +3.932305 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5410 | co(екс.) = 70.4925 |
| | pr = -45.8358 +3.5579 * co -0.025236 * co * co | Y(екс.) = 79.5673 | pr(екс.) = 79.5673 |
| Едномерно сечение при X3 = +1, X2 = +1 | | | |
| int = 88.0000 [ c ] | Y = 76.7612 +3.6948 * X1 -3.633999 * X1*X1 | X1(екс.) = 0.5084 | co(екс.) = 70.1004 |
| | pr = -46.3115 +3.5381 * co -0.025236 * co * co | Y(екс.) = 77.7003 | pr(екс.) = 77.7003 |

**One-dimensional intersections by Motivation of 85%**

By Motivation of 85% we can observe the following one-dimensional intersections of the two-dimensional ones.

We can see here three one-dimensional intersections by all possible combinations with
X1 = -1; X1 = 0 and X1 = +1.

| | | | |
|---|---|---|---|
| Едномерно сечение при X1 = -1, X2 = -1 | | | |
| int = 68.0000 [ c ] | Y = 75.5998 +4.2753 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2561 | agr(екс.) = 73.8419 |
| | pr = -120.6319 +5.4781 * agr -0.0371 * agr * agr | Y(екс.) = 76.1473 | pr(екс.) = 76.1473 |
| Едномерно сечение при X1 = -1, X2 = 0 | | | |
| int = 78.0000 [ c ] | Y = 92.183152 +4.400294 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2636 | agr(екс.) = 73.9542 |
| | pr = -117.6400 +5.4865 * agr -0.0371 * agr * agr | Y(екс.) = 85.2331 | pr(екс.) = 85.2331 |
| Едномерно сечение при X1 = -1, X2 = +1 | | | |
| int = 88.0000 [ b ] | Y = 83.0969 +4.5253 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2711 | agr(екс.) = 74.0666 |
| | pr = -119.7795 +5.4948 * agr -0.0371 * agr * agr | Y(екс.) = 83.7103 | pr(екс.) = 83.7103 |
| Едномерно сечение при X1 = -1, X3 = -1 | | | |
| agr = 55.0000 [ t ] | Y = 71.9067 +3.6236 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.3415 | int(екс.) = 81.4154 |
| | pr = -279.0976 +8.6378 * int -0.053047 * int * int | Y(екс.) = 72.5255 | pr(екс.) = 72.5255 |
| Едномерно сечение при X1 = -1, X3 = 0 | | | |
| agr = 70.0000 [ t ] | Y = 92.183152 +3.748571 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.3533 | int(екс.) = 81.5332 |
| | pr = -267.3262 +8.6503 * int -0.053047 * int * int | Y(екс.) = 85.3153 | pr(екс.) = 85.3153 |
| Едномерно сечение при X1 = -1, X3 = +1 | | | |
| agr = 85.0000 [ t ] | Y = 80.7073 +4.5253 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.4265 | int(екс.) = 82.2653 |
| | pr = -277.3304 +8.7279 * int -0.053047 * int * int | Y(екс.) = 81.6724 | pr(екс.) = 81.6724 |

Построяване на едномерно сечение за X1 = 0, X2 = -1

int = 68.0000 [ c ]

Y = 83.1298 +4.2753 * X3 -8.346049 * X3*X3 | X3ек. = 0.2561 | agrек. = 73.8419

pr = -113.1018 +5.4781 * agr -0.0371 * agr * agr | Yек. = 83.6773 | prек. = 83.6773

One dimensional intersection at X1 = 0, X2 = 0

int = 78.0000 [ c ]

Y = 92.183152 +4.400294 * X3 -8.346049 * X3*X3 | X3ек. = 0.2636 | agrек. = 73.9542

pr = -110.1100 +5.4865 * agr -0.0371 * agr * agr | Yек. = 92.7631 | prек. = 92.7631

Построяване на едномерно сечение за X1 = 0, X2 = +1

int = 88.0000 [ b ]

Y = 90.6270 +4.5253 * X3 -8.346049 * X3*X3 | X3ек. = 0.2711 | agrек. = 74.0666

pr = -112.2495 +5.4948 * agr -0.0371 * agr * agr | Yек. = 91.2404 | prек. = 91.2404

Построяване на едномерно сечение за X1 = 0, X3 = -1

agr = 55.0000 [ t ]

Y = 79.4368 +3.6236 * X2 -5.304741 * X2*X2 | X2ек. = 0.3415 | intек. = 81.4154

pr = -271.5675 +8.6378 * int -0.053047 * int * int | Yек. = 80.0556 | prек. = 80.0556

Построяване на едномерно сечение за X1 = 0, X3 = 0

agr = 70.0000 [ t ]

Y = 92.183152 +3.748571 * X2 -5.304741 * X2*X2 | X2ек. = 0.3533 | intек. = 81.5332

pr = -259.7961 +8.6503 * int -0.053047 * int * int | Yек. = 92.8454 | prек. = 92.8454

Построяване на едномерно сечение за X1 = 0, X3 = +1

agr = 85.0000 [ t ]

Y = 88.2374 +4.5253 * X2 -5.304741 * X2*X2 | X2ек. = 0.4265 | intек. = 82.2653

pr = -269.8003 +8.7279 * int -0.053047 * int * int | Yек. = 89.2025 | prек. = 89.2025

Едномерно сечение при X1 = +1, X2 = -1

int = 68.0000 [ c ]

Y = 83.5869 +4.2753 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2561 | agr(екс.) = 73.8419

pr = -112.6448 +5.4781 * agr -0.0371 * agr * agr | Y(екс.) = 84.1344 | pr(екс.) = 84.1344

Едномерно сечение при X1 = +1, X2 = 0

int = 78.0000 [ c ]

Y = 92.183152 +4.400294 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2636 | agr(екс.) = 73.9542

pr = -109.6529 +5.4865 * agr -0.0371 * agr * agr | Y(екс.) = 93.2202 | pr(екс.) = 93.2202

Едномерно сечение при X1 = +1, X2 = +1

int = 88.0000 [ b ]

Y = 91.0840 +4.5253 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2711 | agr(екс.) = 74.0666

pr = -111.7925 +5.4948 * agr -0.0371 * agr * agr | Y(екс.) = 91.6974 | pr(екс.) = 91.6974

Едномерно сечение при X1 = +1, X3 = -1

agr = 55.0000 [ t ]

Y = 79.8938 +3.6236 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.3415 | int(екс.) = 81.4154

pr = -271.1105 +8.6378 * int -0.053047 * int * int | Y(екс.) = 80.5126 | pr(екс.) = 80.5126

Едномерно сечение при X1 = +1, X3 = 0

agr = 70.0000 [ t ]

Y = 92.183152 +3.748571 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.3533 | int(екс.) = 81.5332

pr = -259.3391 +8.6503 * int -0.053047 * int * int | Y(екс.) = 93.3024 | pr(екс.) = 93.3024

Едномерно сечение при X1 = +1, X3 = +1

agr = 85.0000 [ t ]

Y = 88.6944 +4.5253 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.4265 | int(екс.) = 82.2653

pr = -269.3433 +8.7279 * int -0.053047 * int * int | Y(екс.) = 89.6595 | pr(екс.) = 89.6595

We can see here three one-dimensional intersections by all possible combinations with X2 = -1; X2 = 0 and X2 = +1.

Едномерно сечение при X2 = -1, X1 = -1

| | | | |
|---|---|---|---|
| co = 52.0000 [ pr ] | Y = 74.0765 +4.2753 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2561 | agr(екс.) = 73.8419 |
| | pr = -120.6319 +5.4781 * agr -0.0371 * agr * agr | Y(екс.) = 74.6240 | pr(екс.) = 74.6240 |

Едномерно сечение при X2 = -1, X1 = 0

| | | | |
|---|---|---|---|
| co = 64.0000 [ pr ] | Y = 92.183152 +4.400294 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2636 | agr(екс.) = 73.9542 |
| | pr = -119.1633 +5.4865 * agr -0.0371 * agr * agr | Y(екс.) = 83.7098 | pr(екс.) = 83.7098 |

Едномерно сечение при X2 = -1, X1 = +1

| | | | |
|---|---|---|---|
| co = 76.0000 [ pr ] | Y = 81.5737 +4.5253 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2711 | agr(екс.) = 74.0666 |
| | pr = -121.3028 +5.4948 * agr -0.0371 * agr * agr | Y(екс.) = 82.1871 | pr(екс.) = 82.1871 |

Едномерно сечение при X2 = -1, X3 = -1

| | | | |
|---|---|---|---|
| agr = 55.0000 [ t ] | Y = 70.3835 +3.8685 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.5469 | co(екс.) = 70.5633 |
| | pr = -50.8436 +3.4660 * co -0.024559 * co * co | Y(екс.) = 71.4414 | pr(екс.) = 71.4414 |

Едномерно сечение при X2 = -1, X3 = 0

| | | | |
|---|---|---|---|
| agr = 70.0000 [ t ] | Y = 92.183152 +3.993549 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.5646 | co(екс.) = 70.7754 |
| | pr = -38.7640 +3.4764 * co -0.024559 * co * co | Y(екс.) = 84.2572 | pr(екс.) = 84.2572 |

Едномерно сечение при X2 = -1, X3 = +1

| | | | |
|---|---|---|---|
| agr = 85.0000 [ t ] | Y = 79.1841 +4.1185 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.5823 | co(екс.) = 70.9874 |
| | pr = -43.3764 +3.4868 * co -0.024559 * co * co | Y(екс.) = 80.3832 | pr(екс.) = 80.3832 |

Едномерно сечение при X2 = 0, X1 = -1

| | | | |
|---|---|---|---|
| co = 54.0000 [ b ] | Y = 84.6531 +4.2753 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2561 | agr(екс.) = 73.8419 |
| | pr = -111.5786 +5.4781 * agr -0.0371 * agr * agr | Y(екс.) = 85.2006 | pr(екс.) = 85.2006 |

Едномерно сечение при X2 = 0, X1 = 0

| | | | |
|---|---|---|---|
| co = 64.0000 [ b ] | Y = 92.183152 +4.400294 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2636 | agr(екс.) = 73.9542 |
| | pr = -110.1100 +5.4865 * agr -0.0371 * agr * agr | Y(екс.) = 92.7631 | pr(екс.) = 92.7631 |

Едномерно сечение при X2 = 0, X1 = +1

| | | | |
|---|---|---|---|
| co = 74.0000 [ b ] | Y = 92.6402 +4.5253 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2711 | agr(екс.) = 74.0666 |
| | pr = -110.2363 +5.4948 * agr -0.0371 * agr * agr | Y(екс.) = 93.2536 | pr(екс.) = 93.2536 |

Едномерно сечение при X2 = 0, X3 = -1

| | | | |
|---|---|---|---|
| agr = 55.0000 [ t ] | Y = 79.4368 +3.8685 * X1 -3.536538 * X1*X1 | X2(екс.) = 0.5469 | co(екс.) = 70.1477 |
| | pr = -41.7903 +3.4660 * co -0.024559 * co * co | Y(екс.) = 80.4947 | pr(екс.) = 80.4947 |

Едномерно сечение при X2 = 0, X3 = 0

| | | | |
|---|---|---|---|
| agr = 70.0000 [ t ] | Y = 92.183152 +3.993549 * X1 -3.536538 * X1*X1 | X2(екс.) = 0.5646 | co(екс.) = 70.7754 |
| | pr = -78.6996 +5.0277 * co -0.036838 * co * co | Y(екс.) = 93.3106 | pr(екс.) = 93.1527 |

Едномерно сечение при X2 = 0, X3 = +1

| | | | |
|---|---|---|---|
| agr = 85.0000 [ t ] | Y = 88.2374 +4.1185 * X1 -3.536538 * X1*X1 | X2(екс.) = 0.5823 | co(екс.) = 70.9874 |
| | pr = -34.3231 +3.4868 * co -0.024559 * co * co | Y(екс.) = 89.4365 | pr(екс.) = 89.4365 |

Едномерно сечение при X2 = +1, X1 = -1

co = 52.0000 [ b ]

Y = 81.5737 +4.2753 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2561 | agr(екс.) = 73.8419

pr = -114.6580 +5.4781 * agr -0.0371 * agr * agr | Y(екс.) = 82.1212 | pr(екс.) = 82.1212

Едномерно сечение при X2 = +1, X1 = 0

co = 64.0000 [ b ]

Y = 92.183152 +4.400294 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2636 | agr(екс.) = 73.9542

pr = -111.6661 +5.4865 * agr -0.0371 * agr * agr | Y(екс.) = 91.2070 | pr(екс.) = 91.2070

Едномерно сечение при X2 = +1, X1 = +1

co = 76.0000 [ b ]

Y = 89.0708 +4.5253 * X3 -8.346049 * X3*X3 | X3(екс.) = 0.2711 | agr(екс.) = 74.0666

pr = -113.8056 +5.4948 * agr -0.0371 * agr * agr | Y(екс.) = 89.6842 | pr(екс.) = 89.6842

Едномерно сечение при X2 = +1, X3 = -1

agr = 55.0000 [ t ]

Y = 77.8806 +3.8685 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.5469 | co(екс.) = 70.5633

pr = -43.3465 +3.4660 * int -0.024559 * int * int | Y(екс.) = 78.9386 | pr(екс.) = 78.9386

Едномерно сечение при X2 = +1, X3 = 0

agr = 70.0000 [ t ]

Y = 92.183152 +3.993549 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.5646 | co(екс.) = 70.7754

pr = -31.2668 +3.4764 * int -0.024559 * int * int | Y(екс.) = 91.7544 | pr(екс.) = 91.7544

Едномерно сечение при X2 = +1, X3 = +1

agr = 85.0000 [ t ]

Y = 86.6812 +4.1185 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.5823 | co(екс.) = 70.9874

pr = -35.8792 +3.4868 * int -0.024559 * int * int | Y(екс.) = 87.8803 | pr(екс.) = 87.8803

We can see here three one-dimensional intersections by all possible combinations with X3 = -1; X3 = 0 and X3 = +1.

Едномерно сечение при X3 = -1, X1 = -1

co = 52.0000 [ pr ]

Y = 71.9067 +4.1236 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.3887 | int(екс.) = 81.8867

pr = -274.3098 +8.6878 * int -0.0530 * int * int | Y(екс.) = 72.7081 | pr(екс.) = 72.7081

Едномерно сечение при X3 = -1, X1 = 0

co = 64.0000 [ pr ]

Y = 92.183152 +3.748571 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.3533 | int(екс.) = 81.5332

pr = -272.5425 +8.6503 * int -0.0530 * int * int | Y(екс.) = 80.0990 | pr(екс.) = 80.0990

Едномерно сечение при X3 = -1, X1 = +1

co = 76.0000 [ pr ]

Y = 79.8938 +3.3736 * X2 -5.304741 * X2*X2 | X2(екс.) = 0.3180 | int(екс.) = 81.1798

pr = -269.1605 +8.6128 * int -0.0530 * int * int | Y(екс.) = 80.4302 | pr(екс.) = 80.4302

Едномерно сечение при X3 = -1, X2 = -1

int = 68.0000 [ c ]

Y = 70.3835 +4.3685 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.6176 | co(екс.) = 71.4116

pr = -53.5103 +3.5076 * co -0.024559 * co * co | Y(екс.) = 71.7326 | pr(екс.) = 71.7326

Едномерно сечение при X3 = -1, X2 = 0

int = 78.0000 [ c ]

Y = 92.183152 +3.993549 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.5646 | co(екс.) = 70.7754

pr = -42.4570 +3.4764 * co -0.024559 * co * co | Y(екс.) = 80.5642 | pr(екс.) = 80.5642

Едномерно сечение при X3 = -1, X2 = +1

int = 88.0000 [ c ]

Y = 77.8806 +3.6185 * X1 -3.536538 * X1*X1 | X1(екс.) = 0.5116 | co(екс.) = 70.1391

pr = -42.0132 +3.4451 * co -0.024559 * co * co | Y(екс.) = 78.8063 | pr(екс.) = 78.8063

**Едномерно сечение при X3 = 0, X1 = -1**

co = 52.0000 [ b ]

Y = 84.6531 +4.1236 * X2 -5.304741 * X2*X2 — X2(екс.) = 0.3887 — int(екс.) = 81.8867

pr = -261.5635 +8.6878 * int -0.0530 * int * int — Y(екс.) = 85.4544 — pr(екс.) = 85.4544

**Едномерно сечение при X3 = 0, X1 = 0**

co = 64.0000 [ b ]

Y = 92.183152 +3.748571 * X2 -5.304741 * X2*X2 — X2(екс.) = 0.3533 — int(екс.) = 81.5332

pr = -259.7961 +8.6503 * int -0.0530 * int * int — Y(екс.) = 92.8454 — pr(екс.) = 92.8454

**Едномерно сечение при X3 = 0, X1 = +1**

co = 76.0000 [ b ]

Y = 92.6402 +3.3736 * X2 -5.304741 * X2*X2 — X2(екс.) = 0.3180 — int(екс.) = 81.1798

pr = -256.4141 +8.6128 * int -0.0530 * int * int — Y(екс.) = 93.1765 — pr(екс.) = 93.1765

**Едномерно сечение при X3 = 0, X2 = -1**

int = 68.0000 [ c ]

Y = 83.1298 +4.3685 * X1 -3.536538 * X1*X1 — X1(екс.) = 0.6176 — co(екс.) = 70.9959

pr = -40.7640 +3.5076 * co -0.024559 * co * co — Y(екс.) = 84.4789 — pr(екс.) = 84.4789

**Едномерно сечение при X3 = 0, X2 = 0**

int = 78.0000 [ t ]

Y = 92.183152 +3.993549 * X1 -3.536538 * X1*X1 — X1(екс.) = 0.5646 — co(екс.) = 70.7754

pr = -29.7106 +3.4764 * co -0.024559 * co * co — Y(екс.) = 93.3106 — pr(екс.) = 93.3106

**Едномерно сечение при X3 = 0, X2 = +1**

int = 88.0000 [ t ]

Y = 90.6270 +3.6185 * X1 -3.536538 * X1*X1 — X1(екс.) = 0.5116 — co(екс.) = 70.1391

pr = -29.2668 +3.4451 * co -0.024559 * co * co — Y(екс.) = 91.5526 — pr(екс.) = 91.5526

**Едномерно сечение при X3 = +1, X1 = -1**

co = 52.0000 [ b ]

Y = 80.7073 +4.1236 * X2 -5.304741 * X2*X2 — X2(екс.) = 0.3887 — int(екс.) = 81.8867

pr = -265.5092 +8.6878 * int -0.0530 * int * int — Y(екс.) = 81.5087 — pr(екс.) = 81.5087

**Едномерно сечение при X3 = +1, X1 = 0**

co = 64.0000 [ b ]

Y = 92.183152 +3.748571 * X2 -5.304741 * X2*X2 — X2(екс.) = 0.3533 — int(екс.) = 81.5332

pr = -263.7419 +8.6503 * int -0.0530 * int * int — Y(екс.) = 88.8996 — pr(екс.) = 88.8996

**Едномерно сечение при X3 = +1, X1 = +1**

co = 76.0000 [ b ]

Y = 88.6944 +3.3736 * X2 -5.304741 * X2*X2 — X2(екс.) = 0.3180 — int(екс.) = 81.1798

pr = -260.3599 +8.6128 * int -0.0530 * int * int — Y(екс.) = 89.2308 — pr(екс.) = 89.2308

**Едномерно сечение при X3 = +1, X2 = -1**

int = 66.0000 [ c ]

Y = 75.4911 +4.3685 * X1 -3.536538 * X1*X1 — X1(екс.) = 0.6176 — co(екс.) = 71.4116

pr = -44.7097 +3.5076 * co -0.024559 * co * co — Y(екс.) = 80.5332 — pr(екс.) = 80.5332

**Едномерно сечение при X3 = +1, X2 = 0**

int = 78.0000 [ c ]

Y = 92.183152 +3.993549 * X1 -3.536538 * X1*X1 — X1(екс.) = 0.5646 — co(екс.) = 70.7754

pr = -33.6564 +3.4764 * co -0.024559 * co * co — Y(екс.) = 89.3648 — pr(екс.) = 89.3648

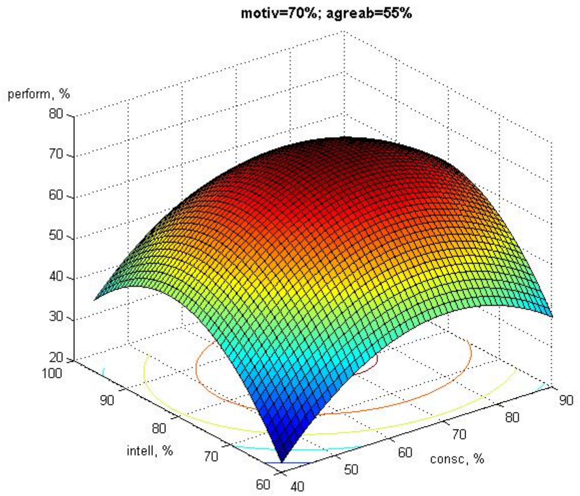
**Едномерно сечение при X3 = +1, X2 = +1**

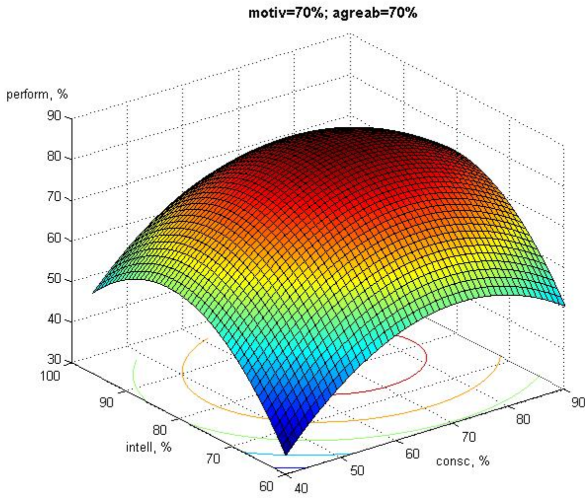
int = 88.0000 [ c ]

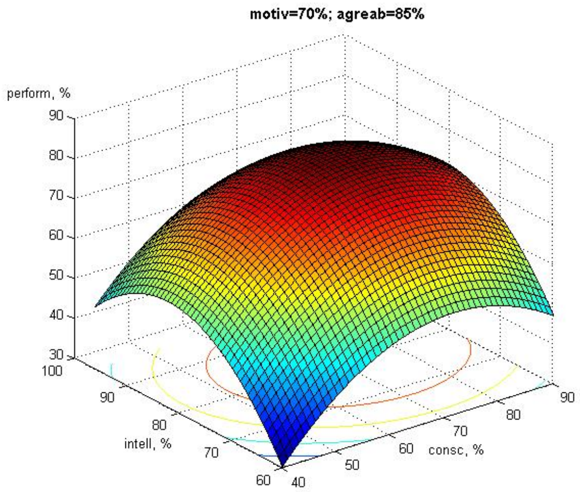
Y = 86.6812 +3.6185 * X1 -3.536538 * X1*X1 — X1(екс.) = 0.5116 — co(екс.) = 70.1391

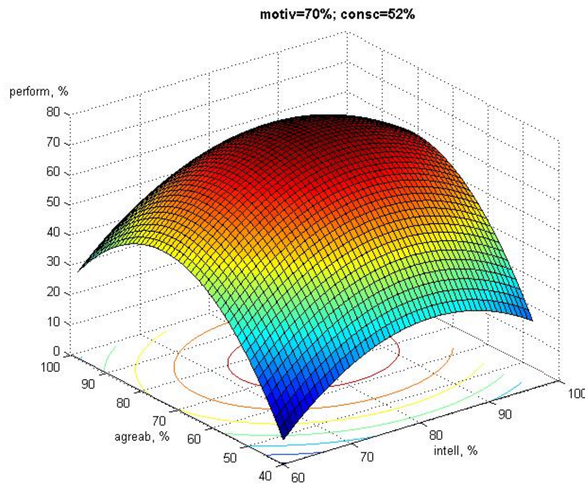
pr = -33.2126 +3.4451 * co -0.024559 * co * co — Y(екс.) = 87.6068 — pr(екс.) = 87.6068

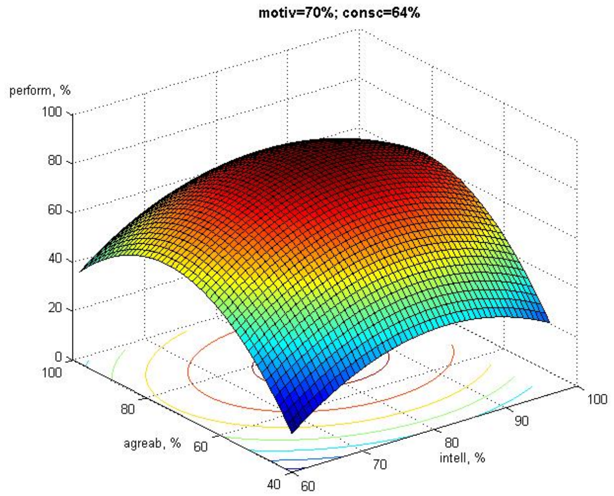
**The response surface graphics by Motivation of 70%**

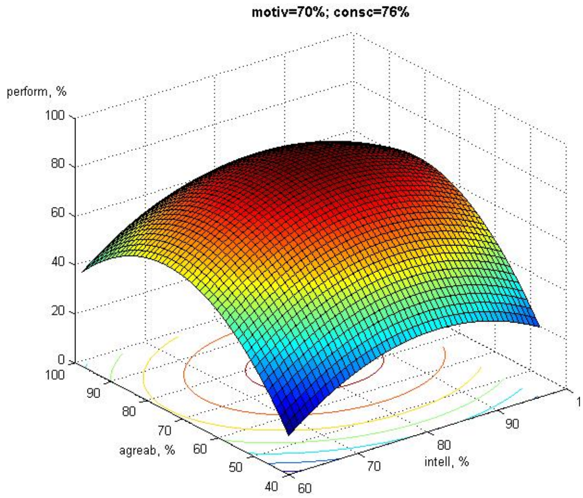


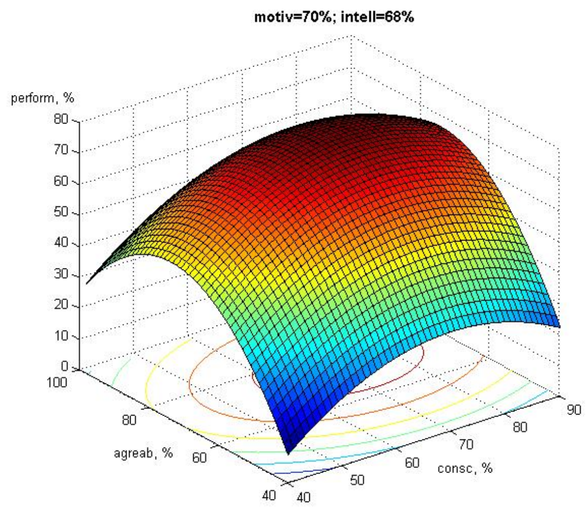



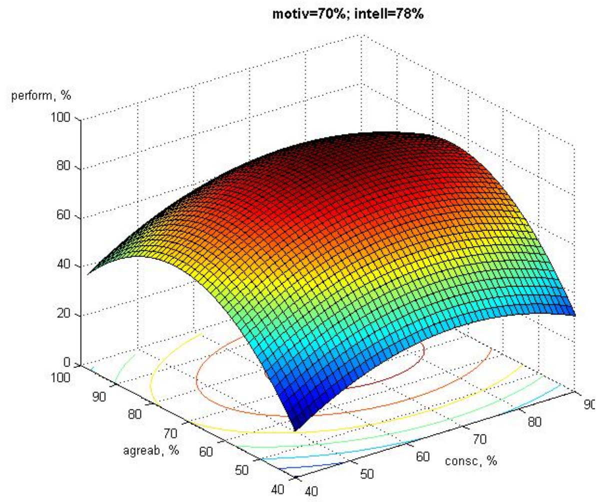



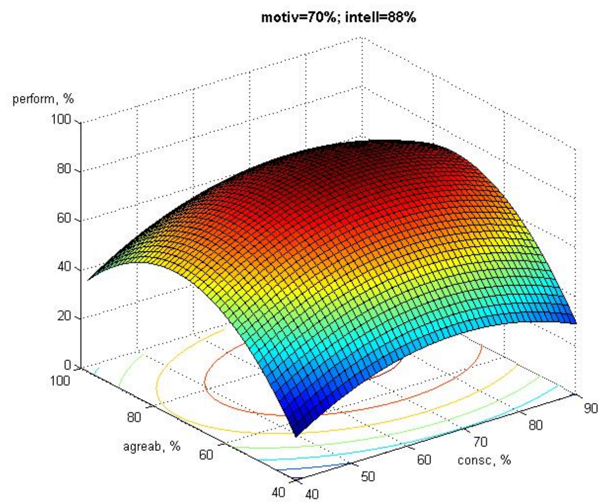



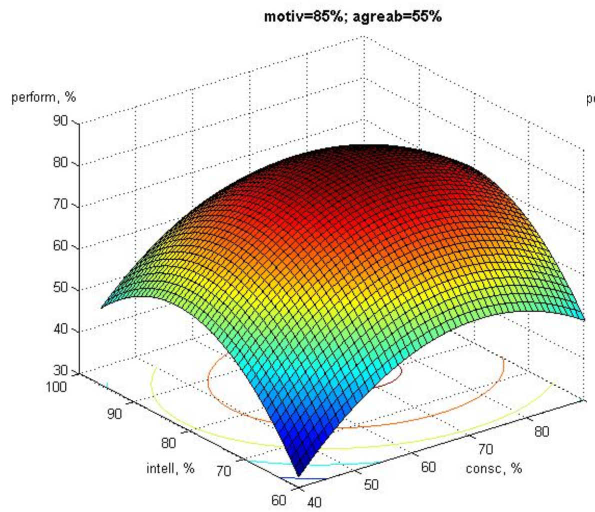



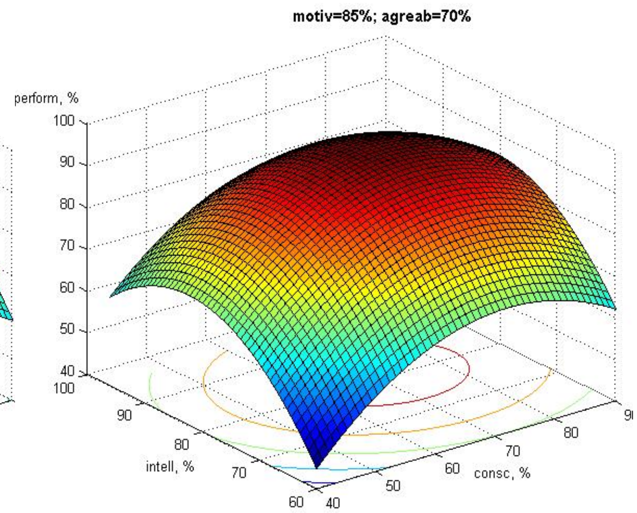


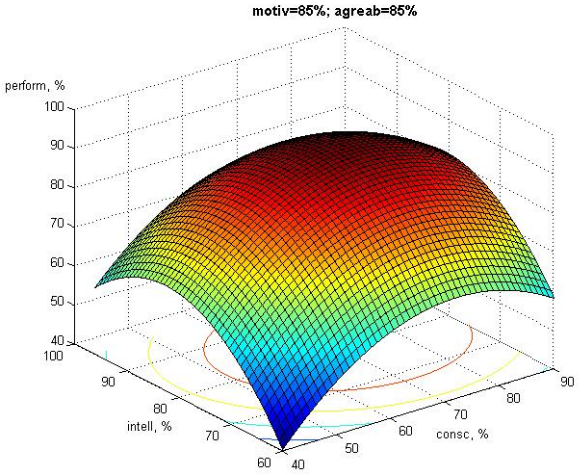



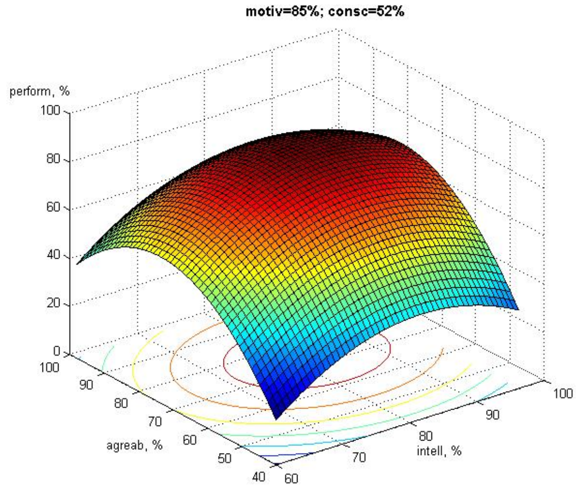



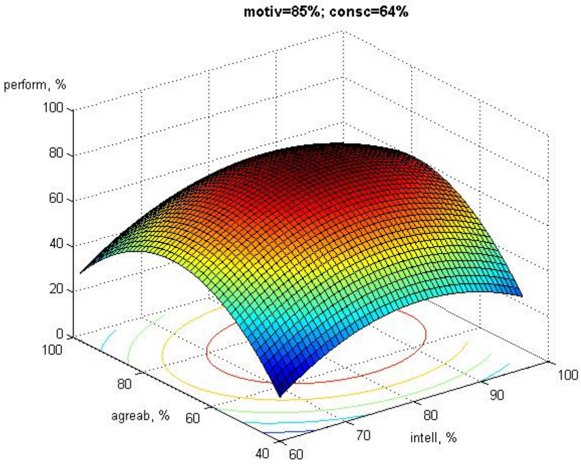

**The response surface graphics by Motivation of 85%**

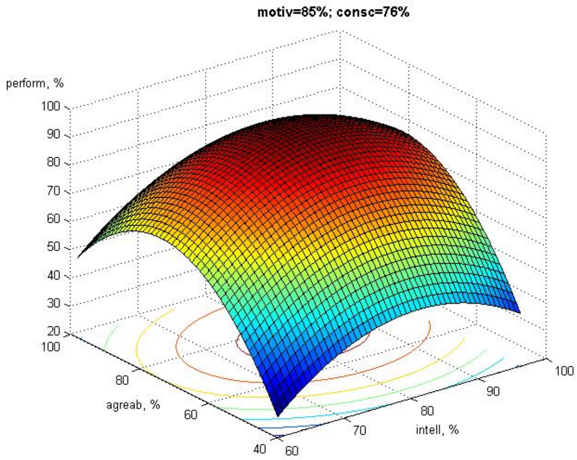


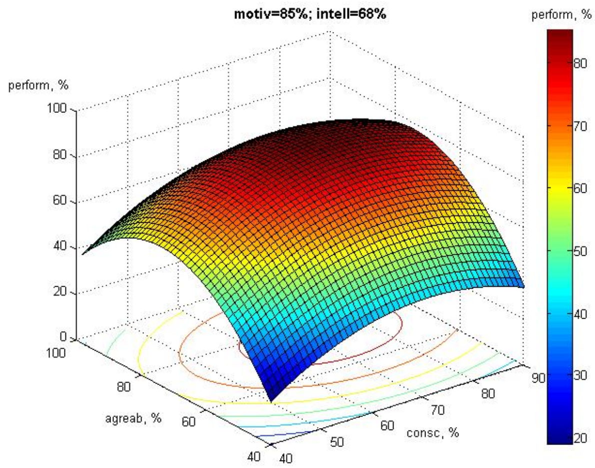



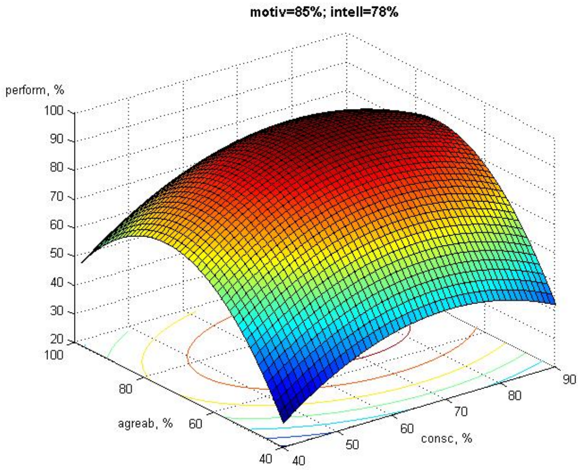

motiv=85%; agreab=85%
perform, %
intell, %
consc, %

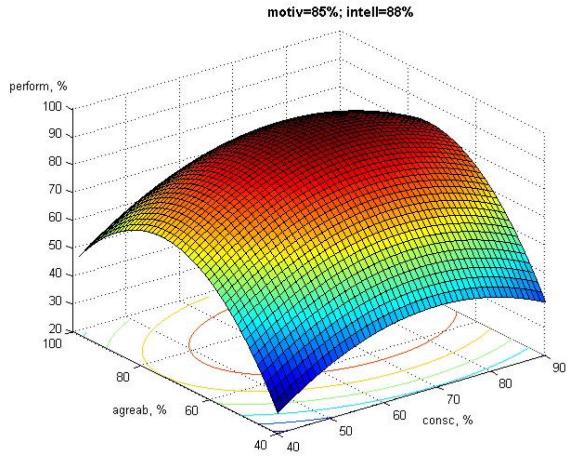

motiv=85%; consc=52%
perform, %
agreab, %
intell, %


motiv=85%; consc=64%
perform, %
agreab, %
intell, %


motiv=85%; consc=76%
perform, %
agreab, %
intell, %


motiv=85%; intell=68%
perform, %
agreab, %
consc, %
perform, %


motiv=85%; intell=78%
perform, %
agreab, %
consc, %




## Complete list of the validation results

The complete list with the 100 validation results from the 'Software Human Factors Test' can be seen on the following figures:

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG FAKULTÄT FÜR INFORMATIK

**Software Human Factors Test**

| Surgency or Extraversion | Agreeableness | Conscientiousness | Emotional Stability | Intellect or Imagination | Motivation | Self estimated performance | Calculated Performance | Difference |
|---|---|---|---|---|---|---|---|---|
| 86% | 90% | 80% | 78% | 76% | Low (~55%) | 62% | 60% | 2% |
| 78% | 68% | 68% | 78% | 82% | Medium (~70%) | 80% | 83% | 3% |
| 64% | 72% | 84% | 64% | 80% | Medium (~70%) | 85% | 80% | 5% |
| 98% | 82% | 62% | 74% | 74% | Low (~55%) | 68% | 65% | 3% |
| 62% | 88% | 70% | 76% | 74% | Low (~55%) | 65% | 63% | 2% |
| 96% | 90% | 84% | 62% | 80% | Medium (~70%) | 70% | 71% | 1% |
| 60% | 68% | 74% | 92% | 68% | Medium (~70%) | 70% | 73% | 3% |
| 78% | 82% | 88% | 64% | 68% | High (~85%) | 68% | 71% | 3% |
| 72% | 94% | 86% | 90% | 74% | Medium (~70%) | 60% | 62% | 2% |
| 82% | 80% | 60% | 92% | 66% | High (~85%) | 70% | 73% | 3% |
| 60% | 86% | 84% | 88% | 82% | Medium (~70%) | 70% | 75% | 5% |
| 96% | 60% | 70% | 60% | 84% | Medium (~70%) | 80% | 77% | 3% |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 82% | 82% | 92% | 72% | 90% | High (~85%) | 70% | 71% | 1% |
| 64% | 74% | 84% | 76% | 84% | High (~85%) | 80% | 85% | 5% |
| 82% | 86% | 74% | 68% | 60% | Medium (~70%) | 56% | 54% | 2% |
| 82% | 94% | 76% | 60% | 74% | High (~85%) | 68% | 70% | 2% |
| 94% | 78% | 74% | 92% | 64% | High (~85%) | 76% | 74% | 2% |
| 92% | 80% | 82% | 98% | 66% | Medium (~70%) | 68% | 68% | 0% |
| 96% | 86% | 92% | 72% | 80% | Medium (~70%) | 72% | 68% | 4% |
| 76% | 80% | 88% | 70% | 64% | High (~85%) | 70% | 66% | 4% |
| 88% | 68% | 80% | 74% | 98% | High (~85%) | 70% | 70% | 0% |
| 86% | 78% | 74% | 86% | 70% | Low (~55%) | 66% | 65% | 1% |
| 92% | 72% | 72% | 76% | 98% | Medium (~70%) | 70% | 69% | 1% |
| 84% | 78% | 64% | 88% | 74% | High (~85%) | 82% | 85% | 3% |
| 82% | 62% | 74% | 86% | 80% | Low (~55%) | 70% | 66% | 4% |
| 94% | 76% | 82% | 62% | 76% | Medium (~70%) | 82% | 80% | 2% |
| 66% | 76% | 86% | 92% | 72% | Low (~55%) | 60% | 62% | 2% |
| 62% | 62% | 86% | 84% | 88% | Medium (~70%) | 68% | 70% | 2% |
| 60% | 66% | 74% | 64% | 96% | Medium (~70%) | 74% | 70% | 4% |
| 72% | 86% | 96% | 86% | 72% | Medium (~70%) | 60% | 59% | 1% |
| 84% | 74% | 94% | 92% | 74% | Low (~55%) | 60% | 57% | 3% |
| 72% | 90% | 64% | 90% | 88% | Medium (~70%) | 70% | 72% | 2% |
| 62% | 74% | 86% | 74% | 90% | High (~85%) | 78% | 79% | 1% |
| 66% | 76% | 68% | 76% | 98% | Medium (~70%) | 70% | 69% | 1% |
| 96% | 88% | 64% | 94% | 78% | Low (~55%) | 65% | 64% | 1% |
| 96% | 78% | 76% | 96% | 98% | Medium (~70%) | 72% | 68% | 4% |
| 74% | 82% | 62% | 80% | 92% | High (~85%) | 75% | 78% | 3% |
| 72% | 70% | 60% | 98% | 78% | Medium (~70%) | 85% | 81% | 4% |
| 90% | 96% | 82% | 72% | 76% | Medium (~70%) | 60% | 63% | 3% |
| 90% | 82% | 62% | 94% | 72% | High (~85%) | 76% | 80% | 4% |
| 72% | 68% | 72% | 76% | 80% | High (~85%) | 85% | 89% | 4% |
| 90% | 68% | 74% | 92% | 86% | High (~85%) | 86% | 87% | 1% |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 90% | 80% | 68% | 94% | 96% | High (~85%) | 76% | 76% | 0% |
| 68% | 72% | 94% | 96% | 68% | High (~85%) | 64% | 68% | 4% |
| 62% | 94% | 72% | 68% | 70% | High (~85%) | 65% | 67% | 2% |
| 84% | 92% | 64% | 76% | 80% | High (~85%) | 78% | 75% | 3% |
| 90% | 74% | 70% | 60% | 62% | High (~85%) | 75% | 71% | 4% |
| 84% | 80% | 70% | 64% | 72% | Medium (~70%) | 75% | 79% | 4% |
| 80% | 68% | 66% | 88% | 70% | Low (~55%) | 68% | 64% | 4% |
| 66% | 74% | 98% | 60% | 88% | High (~85%) | 70% | 68% | 2% |
| 88% | 78% | 64% | 74% | 78% | Medium (~70%) | 78% | 82% | 4% |
| 70% | 88% | 82% | 90% | 70% | High (~85%) | 78% | 73% | 5% |
| 92% | 74% | 72% | 90% | 98% | High (~85%) | 70% | 74% | 4% |
| 94% | 84% | 80% | 74% | 74% | Medium (~70%) | 75% | 76% | 1% |
| 70% | 80% | 80% | 70% | 84% | High (~85%) | 80% | 85% | 5% |
| 70% | 76% | 68% | 66% | 68% | Medium (~70%) | 70% | 75% | 5% |
| 84% | 86% | 82% | 84% | 72% | Medium (~70%) | 70% | 72% | 2% |
| 86% | 70% | 60% | 98% | 88% | High (~85%) | 80% | 84% | 4% |
| 66% | 94% | 74% | 90% | 68% | Low (~55%) | 55% | 51% | 4% |
| 66% | 82% | 98% | 64% | 84% | High (~85%) | 68% | 68% | 0% |
| 72% | 96% | 72% | 70% | 90% | Medium (~70%) | 60% | 63% | 3% |
| 98% | 74% | 70% | 84% | 88% | Medium (~70%) | 80% | 82% | 2% |
| 98% | 92% | 92% | 88% | 74% | High (~85%) | 60% | 63% | 3% |
| 66% | 62% | 78% | 98% | 90% | Medium (~70%) | 72% | 73% | 1% |
| 86% | 62% | 66% | 90% | 94% | Medium (~70%) | 75% | 70% | 5% |
| 66% | 84% | 78% | 60% | 84% | Medium (~70%) | 75% | 79% | 4% |
| 86% | 82% | 86% | 70% | 66% | High (~85%) | 72% | 70% | 2% |
| 90% | 88% | 76% | 80% | 86% | Low (~55%) | 62% | 64% | 2% |
| 96% | 94% | 94% | 76% | 82% | Medium (~70%) | 60% | 57% | 3% |
| 86% | 76% | 70% | 62% | 90% | High (~85%) | 90% | 85% | 5% |
| 64% | 82% | 84% | 92% | 94% | Low (~55%) | 60% | 57% | 3% |
| 72% | 78% | 96% | 64% | 72% | High (~85%) | 66% | 70% | 4% |
| 68% | 84% | 82% | 84% | 62% | High (~85%) | 60% | 64% | 4% |
| 82% | 66% | 76% | 88% | 82% | Medium (~70%) | 78% | 81% | 3% |
| 76% | 64% | 76% | 78% | 90% | High (~85%) | 78% | 81% | 3% |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 96% | 88% | 64% | 70% | 82% | Medium (~70%) | 80% | 76% | 4% |
| 70% | 86% | 80% | 66% | 70% | High (~85%) | 75% | 76% | 1% |
| 94% | 66% | 90% | 74% | 72% | High (~85%) | 70% | 75% | 5% |
| 92% | 76% | 80% | 88% | 86% | High (~85%) | 82% | 86% | 4% |
| 88% | 78% | 70% | 98% | 96% | Medium (~70%) | 74% | 72% | 2% |
| 88% | 92% | 88% | 62% | 76% | High (~85%) | 65% | 68% | 3% |
| 80% | 92% | 62% | 74% | 92% | Medium (~70%) | 62% | 65% | 3% |
| 90% | 86% | 66% | 72% | 90% | High (~85%) | 75% | 79% | 4% |
| 90% | 78% | 82% | 96% | 76% | Medium (~70%) | 84% | 79% | 5% |
| 80% | 70% | 86% | 66% | 94% | High (~85%) | 78% | 74% | 4% |
| 82% | 64% | 76% | 88% | 70% | High (~85%) | 85% | 80% | 5% |
| 66% | 66% | 82% | 64% | 84% | Medium (~70%) | 80% | 78% | 2% |
| 88% | 80% | 80% | 96% | 94% | Low (~55%) | 65% | 60% | 5% |
| 88% | 80% | 60% | 80% | 68% | Low (~55%) | 60% | 59% | 1% |
| 90% | 70% | 66% | 68% | 94% | Low (~55%) | 68% | 63% | 5% |
| 66% | 70% | 80% | 86% | 76% | High (~85%) | 82% | 86% | 4% |
| 90% | 90% | 64% | 88% | 74% | Medium (~70%) | 70% | 71% | 1% |
| 94% | 66% | 72% | 74% | 94% | High (~85%) | 76% | 78% | 2% |
| 78% | 72% | 78% | 84% | 82% | Low (~55%) | 72% | 70% | 2% |
| 68% | 86% | 92% | 90% | 80% | High (~85%) | 76% | 72% | 4% |
| 80% | 94% | 68% | 60% | 90% | Medium (~70%) | 70% | 66% | 4% |
| 66% | 76% | 68% | 68% | 74% | Medium (~70%) | 80% | 81% | 1% |
| 68% | 60% | 60% | 64% | 86% | High (~85%) | 76% | 79% | 3% |
| 70% | 72% | 74% | 82% | 94% | High (~85%) | 84% | 80% | 4% |
| 100% | 94% | 86% | 84% | 78% | High (~85%) | 72% | 68% | 4% |

## List of Acronyms:

*Chapter 2*

SEI – Software Engineering Institute
NN – Neural Networks
PCA – Principal Component Analysis
GA – Genetic Algorithm
BBN – Bayesian Belief Network
SRE – Software Risk Evaluation
RED – Risk in Early Design
IS – Information Systems
FMEA – Failure Mode and Effect Analysis
COTS – Commercial On The Shelf
SPP – Software Production Process
SS – Software System
RM – Risk Management
RA – Risk Assessment
RC – Risk Controlling
MIPS – Million Instructions Per Second
HF – Human Factors
HRF – Human Risk Factors

*Chapter 3*

SE – Software Engineering
CASE – Computer Aided Software Engineering
iCASE – integrated Computer Aided Software Engineering
SP – Software Product
SD – Software Development
SR – Supporting Resources
SAM – Structured Analysis Methods
OOSA – Object-Oriented Software Analysis
CBSE - Component-Based Software Engineering
AOSE - Agent-Oriented Software Engineering
SOSE – Service-Oriented System Engineering
SA/SD – Structured Analysis/Structured Design
HIPO – Hierarchical Input Process Output
UML – Unified Modeling Language
OMT – Object Modeling Technique
OOD – Object-Oriented Design
RDD – Responsibility-Driven Design
HOOD – Hierarchical Object-Oriented Design
OOSA – Object-Oriented Software Analysis
DCOM - Distributed Component Object Model
EJB - Enterprise Java Beans
CURE - COTS Usage Risks Evaluation
B-COTS – Building COTS Software Systems
AAII – Australien AI Institute agent development
AUML – Agent-oriented UML
IMPACT – Imperative Maryland Platform for Agents Collaborating Together

COCOMO – Constructive Cost Model
ISBSG – International Software Benchmark Standards Group
MAS - Multi Agent System
MaSE – Multiagent Systems Engineering
MASSIVE – Multiagent Systems Interactive View Engineering
SODA – Societies in Open and Distributed Agent spaces
SOA – Service Oriented Architecture

(GRID, DESIRE, SPR Knowledge-Plan and QSM SLIM are self-names)

SR – Software Resources
CARE – Computer Aided Re-Engineering
CAME – Computer Assisted Measurement Evaluation
SA – Software Application
SM – Software Measurement
GUI – Graphic User Interface
SWOT – Strengths, Weaknesses, Opportunities and Threats
S/W – Software

*Chapter 4*

RPN – Risk Priority Number
S – Severity
O – Occurrence
D – Detection
UPS – Uninterruptible Power Supply
SFMEA – Software Failure Mode and Effect Analysis
SHF–FMEA – Software Human Factor FMEA
PM – Project Manager
TL – Team Leader
SA – Software Architect
BA – Business Analyst
SD – Software Developer
ST – Software Tester
QE – Quality Engineer

*Chapter 5*

FFM – Five Factor Model
OCEAN – Openness, Conscientiousness, Extroversion, Agreeableness, Neuroticism

*Chapter 6*

DoE – Design of Experiment
co = conscientiousness
int = intelligence
agr = agreeableness
pr = performance

# References:


14971, I 2001, *ISO/FDIS 14971:2001 International Standard: Medical Devices - Application of risk management to medical devices*, International Organization for Standartization.

1540, I 2001, *IEEE Standard for Software Life Cycle Processes - Risk Management*, IEEE.

16085, BI 2006, *Systems and software engineering - Life cycle processes - Risk management*, British Standard.

4360, A 2004, *AS/NZS 4360:2004 Australian Standard: Risk Management*, Standards Australia.

62304, I 2004, *IEC 62304 International Standard: Medical device software - Software life-cycle processes*, International Electronical Commission.

Abran, A & Moore, J 2004, *Guide to the Software Engineering Body of Knowledge - SWEBOK*, IEEE Computer Society, Los Alamitos, California.

Adams, J 1995, *Risk*, London: University College Press.

Alter, S & Ginzberg, M 1978, 'Managing Uncertainty in MIS Implementation', *Sloan Management Review*, vol 20, no. 1, pp. 23-31.

Applegate, L, McFarlan, F & McKenney, J 1996, *Corporate Information Systems Management*, Text and Causes, Irwin.

Asfoura, E., Kassem, G., Georgieva, K. & Dumke, R., 2011. Developing Approach for Conception of Appropriate Business Model for Federated ERP Systems, *International Conference on E-Learning, E-Business, Enterprise Information Systems, & E-Government, WORLDCOMP (EEE)*. Las Vegas Nevada, USA CSREA Press.

Atkinson, A & Donev, A 1992, *Optimum Experimental Designs*, Oxford University Press.

Banerjee, N 1995, 'Utilization of FMEA concept in software lifecycle management', *Transactions on Information and Communications Technologies*, vol 11, no. ISSN 1743-3517.

Barki, H, Rivard, S & Talbot, J 1993, 'Toward an Assessment of Software Development Risk', *Journal of Management Information Systems*, vol 10, no. 2, pp. 203-225.

Baybutt, P 1996, 'Human factors in process safety and risk management: needs for models, tools and techniques', *International Workshop on Human Factors in Offshore Operations, US Mineral Mangement Service*, New Orleans.

Bentley, JE, Bank, W & NC, C 2004, *Software Testing Fundamentals - Concepts, Roles, and Terminology*, viewed 3 February 2011, "http://www2.sas.com/proceedings/sugi30/141-30.pdf"

Berny, J & Townsend, P 1993, 'Macrosimulation of project risks - a practical way forward', *International Journal of Project Management*, vol 11, no. 4, pp. 201-208.

Boehm, B 1988, 'A Spiral Model of Software Development and Enhancement', *IEEE Computer*.

Boehm, B 1991, 'Software Risk Management: Principles and Practices', *IEEE Software*.

Boehm, B, Port, D, Yang, Y & Bhuta, J 2003, *Not All CBS Are Created Equally: COTS-Intensive Project Types*, In LNCS 2580. Springer.

Boehm, B & Ross, R 1989, 'Theory W Software Project Management: Principles and Examples', *IEEE Transactions on Software Engineering*.

Bogue, R 2005, *Breaking Down Software Development Roles*, viewed 19 January 2011, "http://www.developer.com/mgmt/article.php/3515426/Anatomy-of-a-Software-Development-Role-Quality-Assurance.htm"

Bowers, J 1994, 'Data for project risk analyses', *International Journal of Project Management*, vol 12, no. 1, pp. 9-16.

Bowles, J & Hanczaryk, W 2008, 'Threat effects analysis: Applying FMEA to model computer system threats', *Annual Reliability and Maintainability Symposium*, Las Vegas, NV, USA.

Box, G & Draper, N 2007, *Response Surfaces, Mixtures and Ridge Analysis*, Wiley and Sons.

Box, G & Hunter, J 1957, ' Multifactor experimental design for exploring response surfaces', *Annals Math. Stat.*, vol 28, pp. 195-242.

Brownlee, K 1965, *Statistical Theory and Methodology in Science and Engineering, 2nd ed.*, New York, Wiley.

Carr, M, Konda, S, Monarch, I, Ulrich, F & Walker, C 1993, 'Taxonomy-Based Risk Identification', Carnegie Mellon University, Pittsburg PA.

Chapman, C & Ward, S 2002, *Managing Project Risk and Uncertainty: A Constructively Simple Approach to Decision Making*, John Wiley & Sons.

Chapman, C & Ward, S 2003, *Project Risk Management: Processes, Techniques and Insights, 2nd edition* , John Wiley & Sons.

Charette, R 1989, *Software Engineering Risk Analysis and Management*, McGraw-Hill, New York.

Charette, R 1990, *Applications Strategies for Risk Analysis*, McGraw-Hill, New York.

Chee, CL, Vij, V & Ramamoorthy, C 1995, 'Using Influence Diagrams for Software Risk Analysis', *TAI '95: Proceedings of the Seventh International Conference on Tools with Artificial Intelligence*, IEEE Computer Society.

Chittister, C, Kirkpatrick, R & Van Scoy, R 1992, 'Risk Management in Practice', *American Programmer*, vol 5, pp. 30-35.

Chong, Y & Brown, E 2000, *Managing Project Risk: Business Risk Management for Project Leaders*, Prentice Hall.

Chrissis, M, Konrad, M & Shrum, S 2003, *CMMI: guidelines for process integration and product improvement*, Addison-Wesley Professional.

Chung, L, Nixon, BA, Yu, E & Mylopoulos, J 2000, *Non-Functional Requirements in Software Engineering*, Kluwer Academic Publ.

CMMI 2002, 'CMMI for Systems Engineering/Software Engineering/Integrated Product and Process Development', CMU/SEI-2002-TR-004, Software Engineering Institute, Carnegie Mellon University, Pittsburg PA.

Cockburn, A 1997, *Surviving Object-Oriented Projects*, Addison-Wesley.

Cohen, J 1988, *Statistical power analysis for the behavioral sciences - 2-nd edition*, Routledge Academic.

Cohran, W & Cox, G 1957, *Experimental Designs 2nd ed.*, Wiley, New York.

Conrow, E 2003, *Effective Risk Management: Some Keys to Success, Second Edition*, AIAA (American Institute of Aeronautics & Ast).

Conrow, E & Shishido, P 1997, 'Implementing Risk Management on Software Intensive Projects', *IEEE Software*, vol 14, no. 3, pp. 83-89.

Cooper, D, Grey, S, Raymond, G & Walker, P 2004, *Project Risk Management Guidelines: Managing Risk in Large Projects and Complex Procurements*, John Wiley & Sons.

Costa, P. T.& McCrae, R.R. 1992, 'Revised NEO Personality Inventory (NEO-PI-R) and NEO Five-Factor Inventory (NEO-FFI). Professional manual', Odessa, FL.: *Psychological Assessment Resources*.

Cox, D 1957, *Planning of Experiments*, New York, Wiley.

Daughtrey, T 2001, *Fundamental Concepts for the Software Quality Engineer* , ASQ Quality Press.

Davies, O 1967, *Statistical Methods in Research and Production, 2nd ed.* , London, Griffin; New York, Hafner.

Davis, G 1982, 'Strategies for Information Requirements Determination', *IBM Systems Journal*, vol 21, no. 1.

Davis, AM 1995, *201 Principles of Software Development*, McGraw Hill Publ.

Dayer 2007, 'Consideration of Human Errors in Risk Management', 2007.

Dean, A & Voss, D 1999, *Design and Analysis of Experiments*, Springer.

Deept, I & Ramanamurthy, NU 2004, 'Effective Risk Management: Risk Analysis Using an Enhanced FMEA Technique', *In Annual Project Management Leadership Conference, India*.

Desmond, CL 2004, *Project Management for Telecommunications Managers*, 1st edn, Kluwer Academic Publishers.

Deursen, T & Kuipers, AV 2003, 'Source-Based Software Risk Assessment', *ICSM '03: Proceedings of the International Conference on Software Maintenance*, IEEE Computer Society.

Dhillon, B 2007, 'Basic Human Reliability and Error Concepts', in *Human Reliability and Error in Transportation Systems*, Springer London.

Digman, J 1990, 'Personality structure: Emergence of the five-factor model', *Annual Review of Psychology 41*, 1990, pp. 417-440.

Dinsmore, PC 2010, *The AMA Handbook of Project Management*, 3rd edn, AMACOM.

Dorofee, A, Walker, J, Alberts, C, Higuera, R, Murray, T & Williams, R 1996, 'Continuous Risk Management Guidebook', Software Engineering Institute, Pittsburg PA.

Dumke, R 2003, *Software Engineering – Eine Einführung für Informatiker und Ingenieure*, (4th edn) Vieweg Publ.

Dumke, R., Schmietendorf, A., Georgieva, K. & Yazbek, H., 2008. Software Measurement for Agile Software Development in Computer Science and Technologies., Volume VI/2 (IEEE Press), pp.42-47.

Dumke, R., Schmietendorf, A., Kunz, M. & Georgieva, K., 2009 a. Software-Metriken für die agile Software-Entwicklung. In *Proceedings of the SQS*. Düsseldorf.

Dumke, R., Yazbek, H., Asfoura, E. & Georgieva, K., 2009 b. A General Model for Measurement Improvement. In *Proceedings of Software Process and Product Measurement, International Conferences IWSM and Mensura*. Amsterdam, The Netherlands Springer-Verlag Berlin Heidelberg.

Dumke, R., Richter, K., Asfoura, E. & Georgieva, K., 2009 c. Process Improvement Using Causal Networks. In *International Conference on Software Engineering Research & Practice (SERP) WORLDCOMP*. Las Vegas Nevada, USA.

Dumke, R., Richter, K., Georgieva, K. & Asfoura, E., 2010. Process Improvement Based on Causal Networks. *Eighth ACIS International Conference on Software Engineering Research, Management and Applications (SERA 2010)*. Montréal, Canada IEEE Computer Society Los Alamitos, California.

Dustin, E 2002, *Effective Software Testing: 50 Specific Ways to Improve Your Testing*, Addison-Wesley Professional, "http://www.informit.com/articles/article.aspx?p=31196"

Ebert, C, Dumke, R, 2007, *Software Measurement*, Springer Verlag

Eslinger, S, Ellis, C, Hoting, S & Walden, G 1993, 'PACE System Risk Analysis: An Application', *Second SEI COnference on Software Risk Management*, Pittsburg PA.

Fairley, R 1994, 'Risk Management for Software Projects', *IEEE Software*.

Fang, K-T, Li, R & Sudjianto, A 2006, *Design and Modeling for Computer Experiments*, Chapman & Hall/CRC, Taylor & Francis Group.

Farooq, A., Georgieva, K. & Dumke, R., 2008 a. Challenges in Evaluating SOA Test Processes. Proceedings of *Software Process and Product Measurement. International Conferences IWSM, MetriKon, and Mensura*. Munich, Germany Springer-Verlag Berlin Heidelberg.

Farooq, A., Georgieva, K. & Dumke, R., 2008 b. A Meta-Measurement Approach for Software Test Processes. In *12th IEEE - International Multitopic Conference (IEEE INMIC)*. Karachi, Pakistan

Farooq, A., Georgieva, K., Schmietendorf, A. & Dumke, R., 2010. A Systematic Method for Identifying Testing Project Risks. In *Proceedings of the International Conference on Quality Engineering in Software Technology (CONQUEST 2010)*. Dresden, Germany.

Figallo, C 1998, *Hosting Web Communities*, John Wiley & Sons, Inc..

Fisher, J 2001, 'USER SATISFACTION AND SYSTEM SUCCESS: CONSIDERING THE DEVELOPMENT TEAM', *Australasian Journal of Information Systems*, vol 9, no. 1.

Fisher, R & Yates, F 1973, 'Statistical Tables for Biological, Agricultural and Medical Research', Oliver and Boyd, Edinburg.

Flouris, T & Yilmaz, AK 2010, 'The Risk Management Framework to Strategic Human Resource Management', *International Research Journal of Finance and Economics*, vol 36.

Galagher, B 1999, *Software Acquisition Risk Management Key Process Area - A Guidebook Version 1.02*, CMU/SEI-99-HB-001. Carnegie Mellon University.

Gaulke, M 2002, *Risikomanagement in IT-Projekten*, Oldenbourg Wissenschaftsverlag.

Gemmer, A & Koch, P 1994, 'Rockwell Case Studies in Risk Management', *Third SEI Conference on Software Risk Management* , Pittsburg PA.

Georgieva, K., Neumann, R. & Dumke, R., 2008. Ontological description of software quality standards in Computer Science and Technologies., Volume VI(Number 2, IEEE Press), pp.48-52.

Georgieva, K.; Farooq, A.; Dumke R.R., 2009 a. A Risk Taxonomy for the Software Testing Process. In: G. Büren, R.R. Dumke: Praxis der Software-Messung - Tagungsband des DASMA Software Metrik Kongresses (MetriKon 2009), Kaiserslautern, Shaker Verlag Aachen, S. 247-260

Georgieva, K.; Farooq, A.; Dumke, R.R., 2009 b. Analysis of the Risk Assessment Methods - A Survey. In: Software Process and Product Measurement. International Conferences IWSM 2009 and Mensura 2009, Amsterdam, The Netherlands, Springer-Verlag Berlin Heidelberg, S. 76-86

Georgieva, K., 2009 c. The Incompleteness of the Risk Assessment Methods, in Softwaretechnik-Trends., Volume 2(29), pp.42-47.

Georgieva, K., 2009 d. Human Factors and Software Development Process. In *International Workshop on Empirical Software Engineering in Practice (IWESEP)*. Osaka, Japan.

Georgieva, K., Dumke, R., Neumann, R. & Farooq, A., 2009 e. Software Measurement Modelling and Improvement. In *Proceedings of the International Conference on Software Engineering Research & Practice (SERP), WORLDCOMP'09*. Las Vegas Nevada, USA.

Georgieva, K., Farooq, A. & Dumke, R., 2009 f. Design Quality of Aspect-Oriented and Object-Oriented Programs - An Empirical Comparison. In *4th International Conference on Software and Data Technologies (ICSOFT)*. Sofia, Bulgaria.

Georgieva, K., Neumann, R. & Dumke, R., 2010 a. Applying Human Error Assessment and Reduction Technique (HEART) in the software development process. *Proceedings of the joined International Conferences on Software Measurement (IWSM/MetriKon/Mensura 2010)*. Stuttgart Shaker Verlag Aachen.

Georgieva, K., Neumann, R. & Dumke, R., 2010 b. The Influence of Personal Features on the Project Success. In *5. Hochschul-Roundtable der CECMG/DASMA, Industrielle und gesellschaftliche Herausforderungen beim flexiblen Sourcing von IT-Projekten/-Dienstleistungen*. Darmstadt Shaker Verlag Aachen.

Georgieva, K., Neumann, R. & Dumke, R., 2010 c. Software Quality Standards and Approaches from Ontological Point of View. *Proceedings of the 7th Software Measurement European Forum (SMEF 2010)*. Rome, Italy Libreria CLUP Soc. Coop. Milano, Italy.

Georgieva, K., 2010 d. Conducting FMEA over the Software Development Process, in Software Engineering Notes., Volume 35 (3, ACM New York), p.35.

Georgieva, K., Neumann, R. & Dumke, R., 2010 e. Psychological-based Measurement of Personnel Performance. *Proceedings of the 2010 International Conference on Software Engineering Research & Practice, WORLDCOMP 2010 (SERP 2010)*. Las Vegas Nevada, USA CSREA Press.

Georgieva, K., Neumann, R. & Dumke, R., 2011 a. Failure Mode and Effect Analysis for the software team capabilities. In *MetriKon 2011 - Praxis der Software-Messung. Tagungsband des DASMA Software Merik Kongresses*. Kaiserslautern Shaker Verlag Aachen.

Georgieva, K., Dumke, R. & Fiegler, A., 2011 b. A mathematical model for prediction of the human performance based on the personal features. *Proceedings of the 2011 International Conference on Software Engineering Research & Practice, WORLDCOMP 2011 (SERP 2011),* Las Vegas Nevada, USA CSREA Press.

Georgieva, K., Neumann, R., Fiegler, A. & Dumke, R., 2011 c. Validation of the model for prediction of the human performance. In *Proceedings of the Joint Conference of the 21st International Workshop on Software Measurement and the 6th International Conference on Software Process and Product Measurement (IWSM-MENSURA 2011)*. Nara, Japan, IEEE Computer Society Los Alamitos, California, Washington, Tokyo.

Goddard, P, Raytheon, SC & Troy, M 2000, 'Software FMEA techniques', *Reliability and Maintainability Symposium*, Los Angeles, CA , USA.

Goldberg, LR 1993, 'The structure of phenotypic personality traits', *American Psychologist* , vol 1, no. 48, p. 26–34.

Grey, S 1995, *Practical Risk Assessment for Project Management* , 1st edn, John Wiley & Sons.

Gupta, D & Sadiq, M 2008, 'Software Risk Assessment and Estimation Model', *International Conference on Computer Science and Information Technology*.

Günther, D., Neumann, R., Georgieva, K. & Dumke, R., 2011. Causal Networks Based Process Improvement. In *Proceedings of the 23rd International Conference on Software Engineering & Knowledge Engineering (SEKE 2011)*. Miami Beach, Florida, USA Printed by Knowledge Systems Institute Illinois.

Hall, E 1998, *Managing Risk: Methods for Software Systems Development*, Addison Wesley, Reading MA.

Hartkopf, S 2004, 'From a Single Discipline Risk Management Approach to an Interdisciplinary One: Adaptation of FMEA to Software Needs', *Eleventh Annual International Workshop on Software Technology and Engineering Practice (STEP'04)*, IEEE.

Hass, KB 2005, *Management Concepts*, viewed 20 January 2011, "http://www.managementconcepts.com/portal/server.pt?in_hi_space=SearchResult&in_hi_control=bannerstart&in_tx_query=the%20business%20analyst"

Hass, KB 2007, *Professioinalizing Business Analysis: Breaking the Cycle of Challenged Projects*, Management Concepts, Inc.

Hefner, R 1994, 'Experience with Applying SEI's Risk Taxonomy', *Third SEI Conference on Software Risk Management SEI*, Pittsburg.

Heldman, K 2009, *Project Management Professional Exam Study Guide*, 5th edn, Willey Publishing.

Higuera, R, Gluch, D, Dorofee, A, Murphy, R, Walker, J & Williams, R 1994, 'An Intoduction to Team Risk Management', Carnegie Mellon University, Pittsburg PA.

Higuera, R & Haimes, Y 1996, 'Software Risk Management', Carnegie Mellon University, Pittsburg PA.

Hillson, D 2004, *Effective Opportunity Management for Projects: Exploiting Positive Risk*, Marcel Dekker.

Hillson, D & Webster, R 2006, 'Managing Risk Attitude using Emotional Literacy', *PMI Global Congress EMEA Proceedings*, Madrid, Spain.

Horn, E & Reinke, T 2002, *Softwarearchitektur und Softwarebauelemente*, Hanser Publ.

*Human Brain* 2011, viewed 19 August 2011, "http://www.drugdevelopment-technology.com/projects/antegren/antegren1.html"

*Human Eye* 2011, viewed 19 August 2011, "http://www.familyconnect.org/images/parentsite/HumanEyeDiagram.jpg"

Humphrey, W 2000, *Introduction to the Team Software Process*, Addison-Wesley.

Humphrey, W 2005, *TSP – Leading a development team*, Addison-Wesley Longman, Amsterdam.

IEEE 1990, 'IEEE Standard Glossary', IEEE Computer Society Press.

*International Personality Item Pool* 1997, viewed 7 March 2011, "http://ipip.ori.org/New_IPIP-50-item-scale.htm"

Islam, S & Dong, W 2008, 'Human factors in software security risk management', *LMSA '08: Proceedings of the first international workshop on Leadership and management in software architecture*, ACM, Leipzig.

ISO 2009, 'ISO/FDIS 31000 Risk management -- Principles and guidelines'.

John, OP, Robins, RW & Pervin, LA 2008, *Handbook of personality: theory and research*, Guilford Press.

Jones, C 1994, *Assessment and Control of Software Risks*, Yourdon Press.

Jones, C 2001, 'Software measurement programs and industry leadership', *STSC CrossTalk 14 (2): 4-7*.

Karolak, D 1996, 'Software Engineering Risk Management', *IEEE*, 1996.

Kasse, T 2004, *Practical Insight into CMMI*, 1st edn, Artech House.

Keil, M, Cule, P, Lyytinen, K & Schmidt, R 1998, 'A Framework for Identifying Software Project Risks', *Communications of the ACM*.

Kendrick, T 2003, *Identifying and Managing Project Risk: Essential Tools for Failure-Proofing Your Project*, American Management Association.

Kerzner, H 2009, *Project Management A Systems Approach to Planning, Scheduling, and Controlling*, 10th edn, John Wiley & Sons, New Jersey.

Khuri, A & Cornell, J 1996, *Response Surfaces:designs and analysis*, Marcel Dekker.

Kim, JW & Jung, W 2003, 'A taxonomy of performance influencing factors for human reliability analysis of emergency tasks', *Journal of Loss Prevention in the Process Industries*, vol 16, no. 6, pp. 479-495.

Klipp, A 2009, *The Software Engineer Job Description Handbook and Career Guide*, Emereo Pty Ltd.

Kontio, J 1997, 'The Riskit Method for Software Risk Management, version 1', CS-TR-3782, Departament of Computer Science, University of Maryland.

Kontio, J 2001, *Software Engineering Risk Management: A Method, Improvement Framework and Empirical Evaluation, PhD Thesis*, Helsinki University of Technology, Helsinki.

Kontio, J & Basili, V 1996, *Risk Knowledge Capture in the Riskit Method*, In SEL Software Engineering Workshop.

Kurble, KE 2008, *The Making of Information Systems Software Engineering and Management in a Globalized World*, 1st edn, Springer.

Laporte, CY, Doucet, M, Bourque, P & Belkébir, Y 2007, 'Utilization of a Set of Software Engineering Roles for a Multinational Organization', *Product-Focused Software Process Improvement 8th International conference PROFES 2007*, Springer-Verlag Berlin Heidelberg, Riga, Latvia.

Lauritsen, T & Stalhane, T 2005, 'Safety Methods in Software Process Improvement', *Software Process Improvement 12th European Conference, EuroSPI 2005*, Springer Berlin / Heidelberg, Budapest, Hungary.

Lyytinen, K 2000, 'A Source Based Questionnaire of Main Software Risks', Technical Report Daimler Chrysler, Ulm, Germany.

Lyytinen, K, Mathiassen, L & Ropponen, J 1996, 'A Framework for Software Risk Management', *Journal of Information Systems*, vol 11, no. 4, pp. 275-285.

Lyytinen, K, Mathiassen, L & Ropponen, J 1998, 'Attention Shaping and Software Risk - A Categorical Analysis of Four Classical Risk Management Approaches', *Information Systems Research*, vol 9, no. 3, pp. 233-255.

Maciaszek, LA 2001, *Requirements Analysis and System Design – Development Informatik Systems with UML*, Addison Wesley Publ.

Mäckel, O 2006, *Software FMEA, Opportunities and benefits of FMEA in the development process of software-intensive technical systems*, viewed 15 February 2011, "http://www.fmeainfocentre.com/papers/"

Marciniak, JJ 1994, *Encyclopedia of Software Engineering*, Vol. I and II edn, John Wiley & Sons Inc.,.

Maslow, A 1987, *Motivation and personality*, Harper Collins, NY, USA.

Mason, RL, Gunst, RF & Hess, JL 2003, *Statistical Design and Analysis of Experiments with Applications to Engineering and Science*, John Wiley & Sons.

McComb, D & Smith, J 1991, 'System Project Failure: The Heuristics of Risk', *Journal of Information Systems Management*, vol 8, no. 1.

McConnell, S 1993, *Code Complete*, Microsoft Press.

McConnell, S 1996, *Rapid Development: Taming Wild Software Schedules*, Microsoft Press.

McConnell, S 2004, *Professional Software Development*, Addison-Wesley.

McDermott, R, Mikulak, R & Beauregard, M 2009, *The Basics of FMEA*, CRC Press Taylor & Francis Group.

McDougall, W 1932, 'Of the words character and personality', *Character Personality*, vol I, pp. 3-16.

McFarlan, F 1974, 'Portfolio approach to information systems', *Harvard Bussiness Review*, pp. 142-150.

Meyers, DJ & Trbovich, D 1993, 'One Project's Approach to Software Risk Management', *Second SEI Conference on Software Risk Management*, Pittsburg PA.

Michaels, J 1996, 'Technical Risk Management', 1996.

Mikkelsen, T & Phirego, S 1997, *Practical Software Configuration Management*, Prentice Hall Publ.

Montgomery, DC 2008, *Design and Analysis of Experiments*, John Wiley & Sons Inc.

Moravec, H 1997, 'When will computer hardware match the human brain?', *Journal of Evolution and Technology*, vol 1.

Morin, JM 1993, 'Risk Driven Project Management: A Practical Approach', *Second SEI Conference on Software Risk Management*, Pittsburg PA.

Moynihan, T 1997, 'How Experienced Project Managers Assess Risk', *IEEE Software*, vol 14, no. 3, pp. 35-41.

Moynihan, T 2002, *Coping with IS/IT Risk Management: The Recipes of Experienced Project Managers*, Springer.

MSF 2002, 'MSF Risk Management Discipline v.1.1, Microsoft Solutions White Paper', Microsoft Corp.

Mulcahy, R 2003, *Risk Management, Tricks of the Trade for Project Managers*, Rmc Publications.

Myers, RH 1971, *Response Surface Methodology*, Boston: Allyn and Bacon, Inc.

Myers, RH, Montgomery, DC & Cook, C 2009, *Response Surface Methodology Process and Product Optimization Using Designed Experiments*, Wiley, New York.

Neumann, P 1985, *The Risks Digest*, viewed 1 September 2011, "http://catless.ncl.ac.uk/Risks/26.54.html" \l "subj1.1"

Neumann, DE 2002, 'An Enhanced Neural Network Technique for Software Risk Analysis', *IEEE Trans. Softw. Eng.*, vol 28, pp. 904--912.

Neumann, R., Georgieva, K. & Dumke, R., 2010 a. Recruiting Excellence for Global Players - How the Most Successful Software Company on Earth Sources Talent. In CECMG/DASMA, *Industrielle und gesellschaftliche Herausforderungen beim flexiblen Sourcing von IT-Projekten/-Dienstleistungen*. Darmstadt Shaker Verlag Aachen.

Neumann, R., Georgieva, K. & Dumke, R., 2010 b. Down-top Enterprise Application Development. In *5. Workshop Bewertungsaspekte serviceorientierter Architekturen (BSOA)*. Karlsruhe Shaker Verlag Aachen.

Neumann, R., Georgieva, K., Dumke, R. & Schmietendorf, A., 2011 a. Reverse Commerce - Adding Information System Support for Customer-centric Market Coordination. In *Proceedings of the Fifth International Conference on Digital Society (ICDS 2011)*. Gosier, Guadeloupe IARIA.

Neumann, R., Georieva, K., Dumke, R. & Schmietendorf, A., 2011 b. Moving E-Commerce Towards E-Commodity - A Consequence of Cloud Computing. In *Proceedings of the Fifth International Conference on Digital Society (ICDS 2011)*. Gosier, Guadeloupe, France IARIA.

Nguyen, D 2001, 'Failure modes and effects analysis for software reliability', *Reliability and Maintainability Symposium*, Philadelphia, PA , USA.

Nogueira, J, Luqi & Bhattacharya, S 2000, 'A Risk Assessment Model for Software Prototyping Projects', *RSP '00: Proceedings of the 11th IEEE International Workshop on Rapid System Prototyping (RSP 2000)*, IEEE Computer Society.

Nolan, R 1973, 'Managing the Computer Resource: A Stage Hypothesis', *Communications of the ACM*.

Nolan, R 1979, 'Managing the Crises in data processing', *Harvard Bussiness Review*.

Orasanu 1993, *Decision-making in the cockpit*, CA: Academic Press, San Diego.

Ould, M 1990, *Strategies for Software Engineering, The management of risk and quality*, John Wiley & Sons.

Ozarin, N & Siracusa, M 2003, 'A process for failure modes and effects analysis of computer software', *Reliability and Maintainability Symposium*.

Palmer, S 1998, *People and Self MANAGEMENT, Team Leader Development Series*, Butterworth-Heinemann.

Pandelios, G, Rumsey, T & Dorofee, A 1996, 'Using Risk Management for Software Process Improvement', *SEPG Conference, Software Engineering Institute*, Pittsburg.

Paul, D, Yeates, D & Hindle, K 2006, *Business Analysis*, British Computer Society.

Pentti, H & Atte, H 2002, ' FAILURE MODE AND EFFECTS ANALYSIS OF SOFTWARE-BASED AUTOMATION SYSTEMS', ISBN951-712-584-4, VTT Industrial Systems.

Perry, WE 2006, *Effective Methods for Software Testing, Third Edition*, John Wiley & Sons.

Pfleeger, SL 1998, *Software Engineering – Theory and Practice*, Prentice-Hall Publ.

PMBOK 2004, *A Guide to the Project Management Body of Knowledge, Third Edition*, Project Management Institute.

PMI 2008, *A guide to the project management body of knowledge*, 4th edn, Project Management Inst.

Pressman, R 2004, *Software Engineering: A Practitioner's Approach* , McGraw-Hill, 6-th edition.

Pritchard, C 2001, *Risk Management: Concepts & Guidance - 2nd edition*, ESI Intl.

Rasmussen 1982, 'Human errors: A taxonomy for describing human malfunction in industrial installations', *Journal of Occupational Accidents*.

Reason 1990, *Human Error*, New York: Cambridge, New York.

Ropponen, J & Lyytinen, K 1997, 'Can software risk management improve system development: an exploratory study', *European Journal of Information Systems*, vol 6, pp. 41-50.

Ropponen, J & Lyytinen, K 2000, 'Components of Software Development Risk: How to Address them? A Project Manager Survey', *IEEE Transactions on Software Engineering*.

Rozanski, N & Woods, E 2005, *Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives*, Addison-Wesley Longman, Amsterdam.

Russell, M & Karol, D 1994, '16PF Fifth Edition administrator's manual.'' Champaign, IL'", Institute for Personality & Ability Testing.

Saucier, G & Ostendorf, F 1999, 'Hierarchical subcomponents of the Big Five personality factors: A cross-language replication', *Journal of Personality and Social Psychology*, no. 76, p. 613–627.

Schmietendorf, A 2009, *Compliance and Risk Management in the Context of Complex and High-Integrated Business Architectures (German)*, Shaker Publ, Aachen, Germany,.

Schoitsch, E & Redmill, F 2006, ' EWICS TC7 Guidelines for the Project Management of the Development of Critical Computer Systems, Standards. ISA ', *The Instrumentation, Systems and Automation Society USA*.

Schuyler, J 2001, *Risk and Decision Analysis in Projects*, 2nd edn, Project Management Institute.

SEI 2011, *SEI*, viewed 1 July 2011, "http://www.sei.cmu.edu/architecture/research/previousresearch/duties.cfm"

Shappell, SA 2000, 'The Human Factors Analysis and Classification System – HFACS'.

Shivhare, M & McCreath, G 2010, 'Practical Considerations for DoE Implementation in Quality By Design', *BioProcess International*, vol 8, no. 6, p. 22–30.

Sisti, F & Joseph, S 1994, 'Software Risk Evaluation Method', Carnegie Mellon University, Pittsburg PA.

Skyttner, L 2005, 'General Systems Theory - Problems, Perspectives, Practice', *World Scientific Publication*, New Jersey.

Smith, P & Merritt, G 2002, *Proactive Risk Management: Controlling Uncertainty in Product Development*, Productivity Press Inc.

Smolders, KCHJ, de Kort, YAW, Kaiser, FG & Tenner, AD 2009, 'Need for Recovery in Offices: Behavior-based Assessment', *Unpublished manuscript, in preparation*.

Sodhi, J & Sodhi, P 2001, *IT Project Management Handbook*, Management Concepts.

Sommerville, I 2004, *Software Engineering* , Addison-Wesley, 7th edition.

Sommerville, I 2008, *Software Engineering*, 8th edn, Addison Wesley.

Soto, CJ & John, OP 2008, 'Measuring Big Five domains and 16 facets using the California Psychological Inventory. Manuscript submitted for publication.'.

Stamatis, D 1995, *Failure Mode and Effect Analysis*, ASQC Quality Press, Wisconsin.

Thayer, R & Dorfman, M 2002, *Software Engineering Volume 1: The Development Process*, IEEE Computer Society/Wiley/Jossey Bass.

Thomsett, R 1992, *Third Wave Project Management*, Prentice Hall.

Tiwana, R & Keil, M 2004-2005, ' The One-Minute Risk Assessment Tool', *Programming Languages*, vol 2, no. 9.

Tupes, E & Cristal, R 1961, 'Recurrent Personality Factors Based on Trait Ratings', Technical Report ASD-TR-61-97, Lackland Air Force Base, TX: Personnel Laboratory, Air Force Systems Command.

Van Scoy, R 1992, 'Software Development Risk:Opportunity, Not Problem', Carnegie Mellon University, PittsburgPA.

Vose, D 2008, *Risk Analysis: A Quantitative Guide*, 3rd edn, John Wiley & Sons.

Vucovich, JP, Stone, RB, Liu, X & Tumer, IY 2007, 'Risk Assessment in Early Software Design Based on the Software Function-Failure Design Method', *COMPSAC '07: Proceedings of the 31st Annual International Computer Software and Applications Conference*, IEEE Computer Society.

Wang, Y 2005, 'On Cognitive Properties of Human Factors in Engineering', Theoretical and Empirical Software Engineering Research Centre, Calgary.

Wang, Y 2008, 'On Cognitive Properties of Human Factors and Error Models in Engineering and Socialization', *Journal of Cognitive Informatics and Natural Intelligence*, vol 2, no. 4, pp. 70-84.

Waring, A & A.I., G 1998, *Managing Risk: Critical issue for survival and success into the 21st century*, International Bussines Press.

Watkins, J 2004, *Testing IT: An Off-The-Shelf Software Testing Process*, Cambridge University Press.

Wideman, R 1998, *Project and Program Risk Management: A Guide to Managing Project Risks and Opportunities*, Project Management Institute.

Williamson, J 1994, 'Experiences with an Independent Risk Assessment Team', *Third SEI Conference on Software Risk Management*, Pittsburg PA.

Williams, RC, Pandelios, GJ & Behrens, SG 1999, 'Software Risk Evaluation Method Description (version 2)', CMU/SEI-99-TR-029, ESC-TR-99-029, Software Engineering Institute.

Yacoub, SM & Ammar, HH 2002, 'A Methodology for Architecture-Level Reliability Risk Analysis', *IEEE Trans. Softw. Eng.*, vol 28, pp. 529--547.

Yanyan, Z & Renzuo, X 2008, 'The Basic Research of Human Factor Analysis Based on Knowledge in Software Engineering', *International Conference on Computer Science and Software Engineering*, IEEE.

Yong, H, Juhua, C, Zhenbang, R, Liu, M & Kang, X 2006, 'A Neural Networks Approach for Software Risk Analysis', *ICDMW '06: Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*, IEEE Computer Society.

Young, H, Juhua, C, Huang, J, Liu, M & Xie, K 2007, 'Analyzing Software System Quality Risk Using Bayesian Belief Network', *GRC '07: Proceedings of the 2007 IEEE International Conference on Granular Computing*, IEEE Computer Society.

*Zoho Challenge* 2010, viewed 6 July 2011, "https://challenge.zoho.com/dellly"

Zuse, H 1998, *A Framework of Software Measurement*, de Gruyter Publ.