

Czech University of Life Sciences
Faculty of Economics and Management
Department of Information Engineering



Diploma Thesis
Component of Invoicing System

Author: Jan KŘIČKA
Supervisor: assoc. prof. Vojtěch Merunka, MSc., Ph.D.

© 2012 CULS Prague

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Department of Information Engineering

Faculty of Economics and Management

DIPLOMA THESIS ASSIGNMENT

Kříčka Jan

Informatics

Thesis title

Component of Invoicing System

Objectives of thesis

Aim of this diploma thesis is to analyze state of existing invoicing system and implement or suggest solutions leading to attract more registered and paying users. Main areas of research are registration process, statistics, invoice signature implementation, invoice filtering, and free invoicing for non-registered user possibility as a marketing tool.

Methodology

Based on knowledge and analysis of past and current state of invoicing system to perform analysis of current state of invoicing system that leads to implementation or discussion of changes. Recapitulations how efficiently changes were implemented.

Schedule for processing

june 2011 - september 2011: literacy research, tool installation

october 2011 - december 2011: software development, example project

january 2012 - march 2012: thesis completion, text finalization

The proposed extent of the thesis

60 - 80 stran

Keywords

invoice, information system, business process, Ruby on Rails, web development

Recommended information sources

D. Thomas, C. Fowler. Programming Ruby. The Pragmatic Programmers, LCC, United States of America, 2004.

D. A. Black, Ruby for Rails. Manning Publications Co., Greenwich, 2006.

The Diploma Thesis Supervisor

Merunka Vojtěch, doc. Ing., Ph.D.

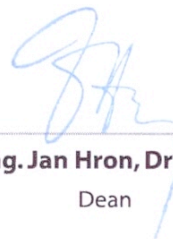
Last date for the submission

November 2012



Ing. Martin Pelikán, Ph.D.

Head of the Department



prof. Ing. Jan Hron, DrSc., dr.h.c.

Dean

Prague November 23. 2012

Statutory declaration

I herewith declare that I have completed the present thesis independently making use only of the specified literature and aids. Sentences or parts of sentences quoted literally are marked as quotations; identification of other references with regard to the statement and scope of the work is quoted. The thesis in this form or in any other form has not been submitted to an examination body and has not been published.

Date: _____ Signature: _____

Acknowledgment

I would like to thank assoc. prof. Vojtěch Merunka, MSc., Ph.D. advisor of my diploma thesis, for guidance and support. Thanks to him I tried new approaches in application development. I would also like to thank external provider of the topic of this thesis GravaStar s.r.o. for their time and support during project development.

Komponenta fakturačního systému

--

Component of invoicing system

Souhrn

V této diplomové práci se zabývám online fakturačním systémem. Systém se jmenuje NuIS a je na trhu od roku 2009. Cílem práce je analýza systému a následný návrh nebo implementace navrženého řešení zaměřené na zvýšení počtu uživatelů systému. Výzkum je zaměřen na oblasti týkajících se neregistrované uživatelů a vnitřních funkcí. Hlavní oblasti výzkumu jsou: registrační proces, fakturace zdarma pro neregistrované uživatele, statistiky, obrázek podpisu faktury, a řazení faktur.

V úvodních kapitolách se zaměřuji na uvedení souvisejících postupů a technologií. V praktické části systém popisuji a analyzuji. Dále popisuji jednotlivé oblasti výzkumu a navrhuji řešení na základě předchozího vyhodnocení systému. Uvádím příklady a výstupy k provedeným implementacím řešení. Na závěr hodnotím výsledky práce.

Summary

In this diploma thesis I am working with online invoicing system. System is named NuIS and it operated on the market since 2009. Aim of this thesis is to analyse the system and design solution or implementation of designed solution targeted on increasing number of the system users. Research is focused on areas considering non-registered users and internal functions. Main areas of research are: registration proces, free invoicing for non-registered user, statistics, invoice signature implementation, and invoice filtering.

In opening chapters I focus on providing information about related techniques and technologies. In project part the system is described the system. Then I describe each area of research and suggest solution based on previous system evaluation. Examples and outputs of implemented solutions are provided. At the end I evaluate the results of the thesis.

Klíčová slova: faktura, informační systém, obchodní proces, Ruby on Rails, vývoj webových aplikací

Keywords: invoice, information system, business proces, Ruby on Rails, web development

Content

1 INTRODUCTION	11
2 GOAL.....	12
3 METHODOLOGY	13
I. LITERATURE RESEARCH.....	14
4 INFORMATION SYSTEM.....	15
4.1 Data, Information and Knowledge	15
4.2 Implementing Information System	16
5 DEVELOPMENT TOOLS	17
5.1 Analysis.....	17
5.1.1 SWOT Analysis.....	17
5.2 Information Technology Infrastructure Library.....	20
5.2.1 Service Design Principles	21
5.2.2 Identifying Business Requirements	24
5.2.3 Service Portfolio Design	24
5.3 Software Quality Requirements and Evaluation	27
5.3.1 Model for External and Internal Software Product Quality	27
5.4 Software Technologies	31
5.4.1 Internet and Web	31
5.4.2 Web Application Environment.....	33
5.4.3 Software Development Environment	38
II. PROJECT.....	40
6 INVOICING SYSTEM	41
6.1 Introduction	41
6.2 System Description.....	41
6.2.1 Invoices Management.....	42
6.2.2 Incomes Management	43
6.2.3 Expenses Management.....	43
6.2.4 Dashboard	43
6.2.5 Companies Management.....	43
6.2.6 Divisions Management	43
6.2.7 Profile Management.....	43
6.3 SWOT Analysis.....	44
6.3.1 External Factors.....	44
6.3.2 Internal Factors.....	46
6.3.3 Evaluation of Internal and External Factors.....	47
6.4 Software Product Quality Evaluation	48

6.4.1 Functional Suitability	48
6.4.2 Reliability	48
6.4.3 Performance Efficiency.....	49
6.4.4 Operability.....	50
6.4.5 Security.....	51
6.4.6 Compatibility	51
6.4.7 Maintainability	51
6.4.8 Portability	52
6.4.9 Software Product Quality Summary.....	52
6.5 Research in Main Areas.....	53
6.5.1 Registration Process.....	53
6.5.2 Free Invoicing	56
6.5.3 Statistics	59
6.5.4 Signature Image Upload	62
6.5.5 Invoice Filtering.....	63
7 CONCLUSION.....	65
8 REFERENCES	67
8.1 Literature.....	67
8.2 Electronic Documents	68
III. ATTACHEMENTS.....	70
9 ATTACHED IMAGES.....	71
10 ATTACHED FILES.....	81
10.1 20110211105948_add_freeze_attributes_to_invoice.rb.....	81

List of figures

Figure 1 External and Internal Analysis [Harvard Business School Press, 2005]	18
Figure 2 The business change process [OGC - Office of Government Commerce – Service Design, 2007].....	21
Figure 3 Project elements in a triangulated relationship [OGC - Office of Government Commerce – Service Design, 2007].....	23
Figure 4 The Service Portfolio [OGC - Office of Government Commerce – Service Design, 2007]	26
Figure 5 Software product quality model [VANICEK, PAPIK, 2012].....	28
Figure 6 Software product quality characteristics and sub characteristics [VANICEK, PAPIK, 2012].....	29
Figure 7 A comparison of OSI and the Internet Architecture models [MILLER, 2009]	32

Figure 8 Client-server architecture [GROVE, 2007]	33
Figure 9 Three-tier architecture [GROVE, 2007]	33
Figure 10 Three-tier architecture with MVC	36
Figure 11 Basic features of NuIS	41
Figure 12 Usage of system	44
Figure 13 Request duration - by mean	49
Figure 14 Example of NuIS GUI	50
Figure 15 Implementation of statistics	61
Figure 16 User's data edit form	62
Figure 17 Invoice filtering implementation	64
Figure 18 System models in UML diagram	71
Figure 19 First invoice - initial state	72
Figure 20 Proposed registration process workflow	73
Figure 21 User part of registration process	74
Figure 22 Company part of registration process - checking ARES	75
Figure 23 Company part of registration process	76
Figure 24 New open invoice form	77
Figure 25 Prepared open invoice	78
Figure 26 Open invoice PDF output	79
Figure 27 Redesigned dashboard	80

List of tables

Table 1 Browser characteristics October 2012 [Browser Statistics]	34
Table 2 SWOT summary	47
Table 3 Core competencies and resources	48
Table 4 Registering process actions	55
Table 5 Actions in OpenInvoiceController	58
Table 6 Statistics indexes definitions	59

1 Introduction

Information and Communication Technologies (ICT) take part in almost everything we do. Companies design their business plan based on utilization of these technologies. ICT help us to communicate and process information more easily and become more efficient.

Computers help us to work with numbers. Their key function is to store and process data, to provide information. For example computers are able to search through company's accounts within less than a second, which could take half an hour if it is done manually. More complex task can be done faster, efficient, and cheaper.

Using ICT can also outsource some of our responsibilities to someone else. This can mean that someone else is dealing with reliability, security, and maintenance of information system or service. ICT can reduce our resources reserved on specific tasks.

Sharing resources and easy collaboration are being used more often. In some projects or business strategies this approach is suitable solution. Instead of designing and implementing our information system we can decide to buy a ready solution. Instead of deploying this system on our own we can decide to use services of company providing network (Internet) access to this computer. In this case requirements on our computer processing power are not so high. On the other side when the network is down we do not have access to the system.

An example of such information system is online invoicing system. I am going to work with real system operated by company GravaStar s.r.o. The idea to create diploma thesis covering system analysis and design emerged at the time when I was working in this company. As the invoicing system is developed aside of client's projects progress is done only while company's capacities are not occupied. I always wanted to try going through design process and implementation by myself. Creating this diploma thesis is the best way how to try that with benefit of being supervised. Having possibility of consulting new approaches, and discovered issues is great advantage.

Areas researched in this thesis could have influence on service of the system. If it is not kept updated technology gets obsolete and the same works for software products. As there is a lot of competition on the open invoicing market the system should have something new to attract customers. This thesis does not aim to change the system to the best product but to design or develop solutions keeping the product attractive. Research areas covers improvements inside the system potentially benefiting registered users and creating new module, component, for not registered users.

2 Goal

This diploma thesis is focused on work with online invoicing system. The main goal is to design solutions for given main areas of research. Main areas of research are:

- Registration process
- Free invoicing for non-registered users as a marketing tool
- Statistics
- Signature image upload
- Invoice filtering

At the beginning system description is going to be provided. The description is going to cover information about the system and its main sections functionality.

After description SWOT analysis is required. Swot analysis is going to reveal internal and external factors that influence business side of the system.

Measuring of software product quality describes system capability and suitability.

Both SWOT analysis and software product quality measurement should reveal parts of the system requiring more attention.

For areas of research initial discussion is provided followed by proposing design of solution on discussed issue. If processed implementation of solution is going to be described.

Achieving all previous tasks should lead to possible increase of incoming new users to the system also usability of the system should be better.

3 Methodology

Before the final version of this diploma thesis assignment was formulated the development team and I did brief SWOT analysis of the system. From discussed areas we chose five areas of research.

Then I consulted suggested areas with my diploma thesis advisor and we agreed on certain approach and formulated the assignment.

System description and introduction was done based on observing and working with the system.

For providing more detailed SWOT analysis I had to find book providing sufficient level of information; mentioned in chapter 5.1.1. Then I did research on the Internet to know the competition state. To get some information about internal state of the system I asked in GravaStar to provide me copy of production database to have access to real data.

For measuring software product quality I used skills and documentation provided in Quality estimation course at school; mentioned in chapter 5.3. Together with copy of production database a obtained copy of production log containing records of system usage.

Production database and production log were processed as confidential data and were evaluated anonymously.

For designing solutions for areas of research I used output of previous analysis and evaluation and consulted approach with system development team. In case of indecision or that I did not wanted to accept suggested approach I went to consult thesis supervisor to have other perspective on the issue. I also used information or knowledge I gained from literature mentioned in 5.2 advised by thesis supervisor.

In implementation process I used previous knowledge of the system, knowledge of web development. Approach of finding proper technique on the Internet was really helpful. So called "Googeling" for technical solution or finding new technology is fast and efficient. I also consulted prepared implementation solutions with the development team leader.

I. Literature research

4 Information System

“Purpose of information systems is to get the right information to the right people at the right time in the right amount and in the right format.” [Rainer, Turban, 2008]. Information system is designed in order to provide useful information. For better understanding we begin by defining three terms: information and another two related terms, data and knowledge.

4.1 Data, Information and Knowledge

Among main goals of information system is to process data economically. To be able to do so under every condition we take a closer look at following.

- *“**Data items** refer to an elementary description of things, events, activities, and transactions that are recorded, classified, and stored but are not organized to convey any specific meaning. Data items can be numbers, letters, figures, sounds, or images.”* [Rainer, Turban, 2008]. Examples of data items can be items assigned to invoice and their prices.
- *“**Information** refers to data that have been organized so that they have meaning and value to the recipient.”* [Rainer, Turban, 2008]. Example of information can be cash flow calculated out of incomes and expenses.
- *“**Knowledge** consists of data and/or information that have been organized and processed to convey understanding, experience, accumulated learning, and expertise as they apply to a current business problem.”* [Rainer, Turban, 2008]. Example of knowledge can be deciding next move on a market, e.g. to introduce new product or wait for current product to reach maturity, based on data and information provided by information system.

I think that understanding these terms is important for understanding information system’s purpose and designing its architecture. As information system is combination of information technology and tasks supporting management and decision making.

Information system consist of five basic resources [O’BRIEN, MARAKAS, 2009] mentions:

- People – IT specialists and end-users
- Hardware – physical aspects of information system
- Software – consists of System, Application, and Utility software
- Data
- Networks – communication media and network support.

4.2 Implementing Information System

Before we decide to implement information system in a company or to support a project we should to ask ourselves several basic questions like they did in [RICHTA, VRANA, 2005]. It is crucial to analyse them carefully in order to provide best results.

Do we really need an information system? But it is not a simple yes or no question more aspects are hidden in it. There are questions like:

- Do we need a better collection of information or representation?
- Will the IS make our company culture better?
- Do we need a better reliability and safety of our information?
- Do we need better support for managing tasks?
- Do we need better way to report to superior institutions?

I think that questions above are not simple polar questions and their outcome is going to be outline of how the information system will look like. An example of such can be case of a company deciding to start using online invoicing system. Initial state of the company is:

- Issuing around 10 invoices per month;
- No information system implemented – using office suite components;
- 6 employees.

Looking closer at some of mentioned questions. Do we need a better collection of information or representation? By asking this question another are going to emerge. How sophisticated are our current methods? How much time is it going to take me if I want to find something from the past? Can someone else understand structure that I am having? Is it necessary for someone to understand it? Answers to these questions reflect our business strategy. Vision of company's future is also going to take part. If company will grow is it still economical, possible, to do things by the old way?

Will the IS make our company culture better? We can provide better environment for our employees. When some new technology is adopted it can make the work more efficient but also make company's employees feel better about it.

5 Development Tools

In order to be able to follow each phase of every project required set of tools, skills, is described.

5.1 Analysis

The Oxford Dictionary defines analysis as “*detailed examination of the elements or structure of something*” [Definition of analysis]. It means that we use analysis to get more detailed information about something. In this paper I am going to analyse a product – online invoicing system NuIS, and a system analysis of this product.

5.1.1 SWOT Analysis

“*As much as some may think that everything devolves from goals, the fact is that practical people form goals based on what is feasible, given the environment in which they must operate their own resources and capabilities.*” [Harvard Business School Press, 2005]. It is important to set up goals to follow but also it is crucial to base them on what is around and inside of our process. Acronym SWOT stands for: Strengths, Weaknesses, Opportunities, and Threads. According to [Harvard Business School Press, 2005] we can define following:

- **Strengths** are capabilities that enable your company or unit to perform well – capabilities that need to be leveraged.
- **Weaknesses** are characteristics that prohibit your company or unit from performing well and need to be addressed.
- **Opportunities** are trends, forces, events, and ideas that your company or unit can capitalize on.
- **Threats** are possible events or forces outside of your control that your company or unit needs to plan for or decide how to mitigate.

Strengths and weaknesses are internal factors and opportunities and threads are external factors. These two categories give us picture of the world in which our company or unit is going to operate. An outcome of combining internal and external analysis should be strategy formulation.

5.1.1.1 External Analysis

As it was said external analysis covers opportunities and threads for company or unit from outside world. List of factors that can influence it is showed in Figure 1.

Important factor are customers without them the business is pointless. Potential and current customers are taken in account. Each of them has specific requirements for our product/service. It is possible to study them and

divide them into groups. As an example users of invoicing system divided into three groups:

- With free account – company makes no profit on them.
- With optimum account – prepaid lower plan.
- With premium account – prepaid highest plan.

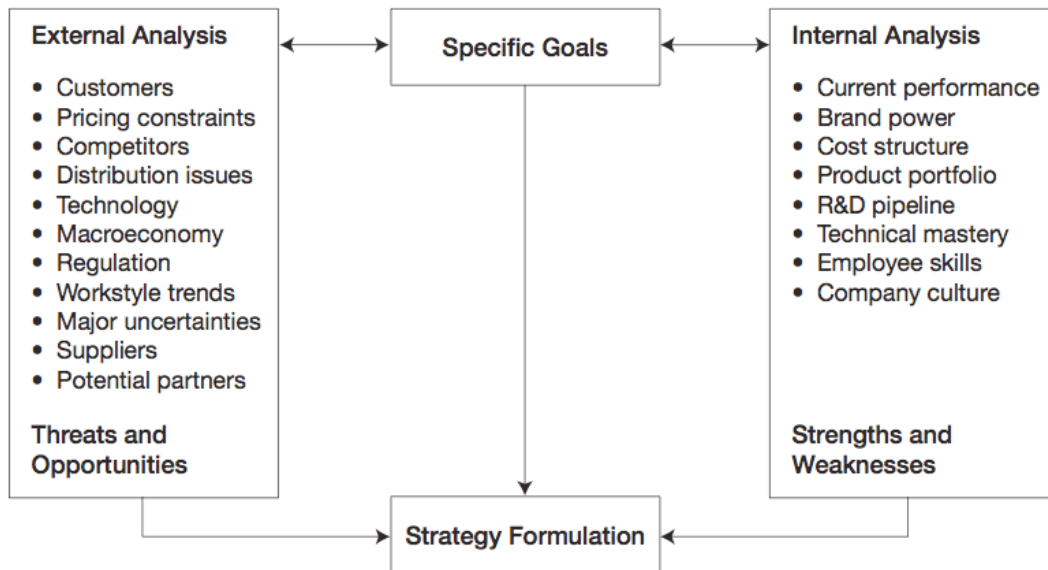


Figure 1 External and Internal Analysis [Harvard Business School Press, 2005]

Only out of users with optimum and premium account the company gets money. As the business plan is to have as many paying users as possible further investigation of customers is necessary. Who are users that switched from free account to prepaid account? Why did they decide to use the system? And to get to know the same for users who used only the free account.

Are customers sensitive to price? It is important to be aware of the relationship between price and customer demand. [Harvard Business School Press, 2005] mentions price elasticity of demand, which is defined as follows:

$$\text{Price elasticity} = \frac{\text{Percentage increase in price}}{\text{Percentage decrease in quality}}$$

Questioners, means of focus groups, and experiments on local market, can determine knowing how customers will react to price change.

Technology can represent both a threat and an opportunity to a company or a unit. New technology can be perceived as thread to our business, e.g. like personal computers and word processing software overtook the place

of typewriters, or how digital cameras replaced photographic film. On other side technology can benefit us. It can provide efficiency, new approaches and satisfy our customers.

5.1.1.2 Internal Analysis

Now lets look inside for company's or unit's strengths and weaknesses. According to [Harvard Business School Press, 2005] there is much to be considered in internal analysis and three of most important strengths and weaknesses should be evaluated: core competencies and processes, financial condition, and management and culture.

*"The term **core competency** refers to a company's expertise or skills in key areas that directly produce superior performance."* [Harvard Business School Press, 2005]. On a question: What are your business core competencies? You should not provide a simple stating answer. For example: "We provide online invoicing system." Instead of that it is better to focus on what are you uniquely good at and what customer values a key activity. To estimate relative power of your core competencies and core processes is benchmarking. "Benchmarking is the process of comparing and measuring your organization against others, anywhere in the world, to gain information on philosophies, practices, and measures that will help your organization take action to improve its performance. In simple terms, benchmarking is the practice of being humble enough to admit that others are better at something and being wise enough to learn how to match, and even surpass, them at it." [COERS, GARDER, 2004].

Core competency should pass following test by [COLLIS, MONTGOMERY, 1995]:

- **Inimitability.** It must be hard to copy. Don't try to base a long-term strategy on something that your competitors can quickly copy.
- **Durability.** Durability refers to the continuing value of the competence or resource. Some brand names have enduring values, e.g. Coca Cola. Some technologies, however, have commercial value for only a few years; then they are swept aside by new and better technologies.
- **Appropriability.** This test determines who captures the value created by your competency or unique resource. In some industries, the lion's share of profits goes to retailers, not to the companies whose ingenuity developed and produced the actual product.
- **Sustainability.** Can your special resource be trumped by a substitute?

- **Competitive superiority.** Is your special competence or resource truly superior to those of competitors? Always rate your strength against best of your rivals.

Core competency should be foundation of new or revised strategy. We can only get meaningful results when we compare company or unit to rivals. I think that it is important to get prepared for every possible situation. It will prevent or minimize further complications and costs. Knowing the project preparation was done well can create more stable environment for its participants.

When internal analysis is ready it is time to assess current financial strength of our company to be able to think of new strategy. New strategy can be costly to implement. [Harvard Business School Press, 2005] advises to provide a full report that includes following:

- **Cash flows.** To what extent are cash flows from current operations sufficient to support a new initiative?
- **Access to outside capital.** If cash flow is insufficient to finance a new strategy, the company will have to look to outside creditors or investors.
- **Other scheduled capital spending plans.** In case that another strategy is planned and has approved capital spending be sure that they will not exceed all available capital level.
- **Hurdle rate of new projects.** The hurdle rate is the minimum rate of return expected from new projects that require substantial capital investments.

5.2 Information Technology Infrastructure Library

Information Technology Infrastructure Library (ITIL) is the most recognized framework for Information Technology Service Management (ITSM) in the world. ITIL bridges the gap between technology and business and it is focused strongly on the processes required delivering effective services to the business customer. Key characteristics of ITIL that contribute to its global success [OGC - Office of Government Commerce, 2007]:

- **Non-proprietary** – ITIL service management practices are applicable in any IT organization because they are not based on any particular technology platform, or industry type
- **Non-prescriptive** – ITIL offers robust, mature and time-tested practices that have applicability to all types of service organizations.
- **Best practice** – ITIL service management practices represent the learning experiences and thought leadership of the world's best in class service providers.

ITIL contains set of concepts and procedures that allow better planning, efficient and improved information technologies used both by suppliers and consumers of services. [OGC - Office of Government Commerce, 2007] defines service: “A **‘service’** is a means of delivering value to customers by facilitating outcomes customers want to achieve without the ownership of specific costs and risks.”

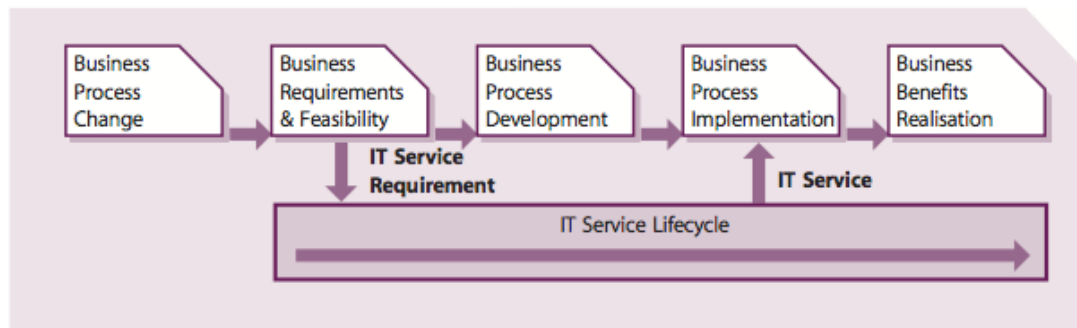


Figure 2 The business change process [OGC - Office of Government Commerce – Service Design, 2007]

5.2.1 Service Design Principles

The role of Service Design stage in overall business change can be defined: “The design of appropriate and innovative IT services, including their architectures, processes, policies and documentation, to meet current and future agreed business requirements.” [OGC - Office of Government Commerce – Service Design, 2007]. Role of IT (Information Technology) within the business process is showed in Figure 2. We can see that IT Service lifecycle starts by defining IT Service Requirement and continues further on, until it gets retired. At point of releasing service to live running difficulties often occur. They recognise following actions that need to be undertaken to ensure that Service Design meets the requirement of business in [OGC - Office of Government Commerce – Service Design, 2007]:

- The new service solution should be added to the overall Service Portfolio from the concept phase, and the Service Portfolio should be updated to reflect the current status
- As part of the initial service/system analysis, there will be need of understanding the Service Level Requirements (SLRs) for the service when it goes live.

- From the SLRs, the Capacity Management team can model this within the current infrastructure to ascertain if this will be able to support the new service.
- If new infrastructure is required for the new service, or extended support, Financial Management will need to be involved to set the budget.

While designing the service we have to consider certain aspects to meet new and evolving business needs. [OGC - Office of Government Commerce – Service Design, 2007] defines:

- **Business process:** to define the functional needs of the service being provided, e.g. telesales, invoicing, orders, credit checking.
- **Service:** the service itself that is being delivered to the customers and business by the service provider, e.g. e-mail, billing.
- **Infrastructure:** all of the IT equipment necessary to delivery the service to the customers and users.
- **Environment:** the environment required securing and operating the infrastructure, e.g. data centres, power, air conditioning.
- **Data:** the data necessary to support the service and provide the information required by the business processes, e.g. customer records, accounts ledger.
- **Applications:** all of the software applications required to manipulate the data and provide the functional requirements of the business processes, e.g. ERM, Financial, CRM.
- **Support Services:** any services that are necessary to support the operation of the delivered service, e.g. a shared service, a managed network service.
- **Operational Level Agreements (OLAs) and contracts:** any underpinning agreements necessary to deliver the quality of service agreed within the SLA.
- **Support Teams:** any internal support teams providing second and third-line support for any of the components required to provide the service, e.g. Unix, mainframe, networks.

For successful design it is necessary to set goals and objectives. I picked some that we should focus on. To design services to satisfy business objectives that can be easily, and efficiently developed and enhanced. To identify and manage risks se you will be able to design secure and resilient IT infrastructures.

Before we identify service requirements it is good to think about balance in out design. We should not only focus on the functional requirements but also on performance. They mention three things in [OGC - Office of Government Commerce – Service Design, 2007]:

- **Functionality:** the service or product and its facilities, functionality and quality, including all of the management and operational functionality required.
- **Resources:** the people, technology and money available.
- **Schedule:** the timescales.

They find this concept very important in [OGC - Office of Government Commerce – Service Design, 2007]. We can see the three elements in balance in Figure 3. *“Changing one side of the triangle invariably has an impact on at least one of the other sides if not both of them. It is vital therefore that the business drivers and needs are fully understood in order that the most effective business solutions are designed and delivered, using the most appropriate balance of these three elements.”* [OGC - Office of Government Commerce – Service Design, 2007].

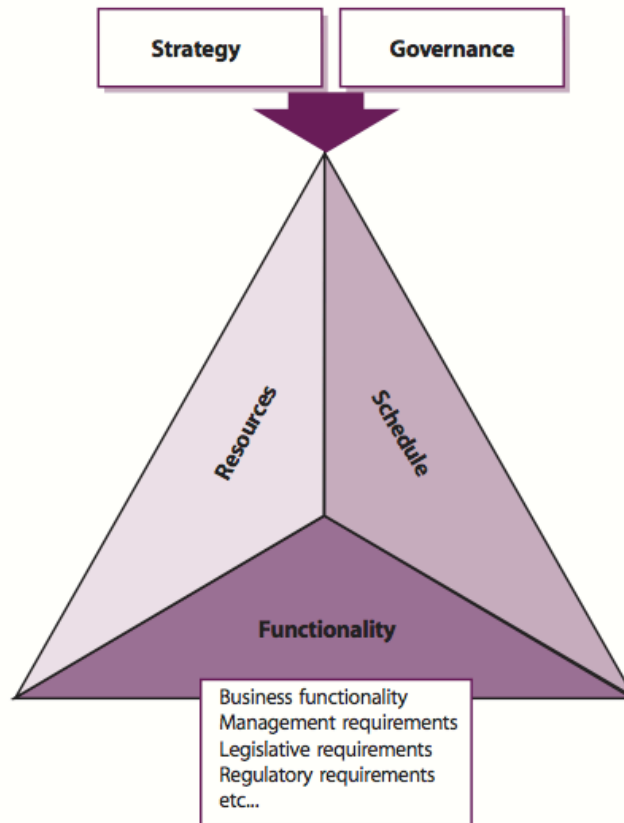


Figure 3 Project elements in a triangulated relationship [OGC - Office of Government Commerce – Service Design, 2007]

5.2.2 Identifying Business Requirements

“In order to design and deliver IT services that meet the needs of the customers and the business, clear, concise, unambiguous specifications of the requirements must be documented and agreed. Time spent in these activities will prevent issues and discussion from arising later with regard to variances from customer and business expectation. This business requirements stage should consist of:

- **Appointment of a project manager**, the creation of a project team and the agreement of project governance by the application of a formal, structured project methodology
- **Identification of all stakeholders**, including the documentation of all requirements from all stakeholders and stakeholder benefits they will obtain from the implementation
- **Requirements analysis**, prioritization, agreement and documentation
- **Determination and agreement** of outline budgets and business benefits. The budget must be committed by management, because it is normal practice to decide next year’s budget in the last quarter of the previous year, so any plans must be submitted within this cycle
- **Resolution** of the potential conflict between business units and agreement on corporate requirements
- **Sign-off processes** for the agreed requirements and a method for agreeing and accepting changes to the agreed requirements. Often the process of developing requirements is an iterative or incremental approach that needs to be carefully controlled to manage ‘scope creep’
- **Development of a customer engagement plan**, outlining the main relationships between IT and the business and how these relationships and necessary communication to wider stakeholders will be managed.” [OGC - Office of Government Commerce – Service Design, 2007].

5.2.3 Service Portfolio Design

*“The **Service Portfolio** is the most critical management system used to support all processes and describes a provider’s services in terms of business value. It articulates business needs and the provider’s response to those needs.” [OGC - Office of Government Commerce – Service Design, 2007]. Service portfolio helps us compare our service competitiveness across alternatives on the market. Definitions of values of business terms correspond to market terms. So then we are able to clarify following strategic questions:*

Why to buy our service? Why from us? How to charge our service? What are the risks and weaknesses? How to allocate resources?

“The Service Portfolio should contain information relating to every service and its current status within the organization. The options of status within the Service Portfolio should include:

- **Requirements:** a set of outline requirements have been received from the business or IT for a new or changed service
- **Defined:** the set of requirements for the new service are being assessed, defined and documented and the SLR is being produced
- **Analysed:** the set of requirements for the new service are being analysed and prioritized
- **Approved:** the set of requirements for the new service have been finalized and authorized
- **Chartered:** the new service requirements are being communicated and resources and budgets allocated
- **Designed:** the new service and its constituent components are being designed – and procured, if required
- **Developed:** the service and its constituent components are being developed or harvested, if applicable
- **Built:** the service and its constituent components are being built
- **Test:** the service and its constituent components are being tested
- **Released:** the service and its constituent components are being released
- **Released:** the service and its constituent components are being released
- **Retired:** the service and its constituent components have been retired.” [OGC - Office of Government Commerce – Service Design, 2007].

Service portfolio contains details of all services and their statuses assigned to current stage in Service Lifecycle, as we can see on Figure 4. From all statuses listed customer can access only services with status chartered or operational.

“Each organization should carefully design its Service Portfolio, the content and the access allowed to the content. The content should include:

- **Service name**
- **Service description**
- **Service status**
- **Service classification and criticality**
- **Applications used**
- **Data and/or data schema used**

- **Business processes supported**
- **Business owners**
- **Business users**
- **IT owners**
- **Service warranty level, SLA and SLR references**
- **Supporting services**
- **Supporting resources**
- **Dependent services**
- **Supporting OLAs, contracts and agreements**
- **Service costs**
- **Service charges (if applicable)**
- **Service revenue (if applicable)**
- **Service metrics.**

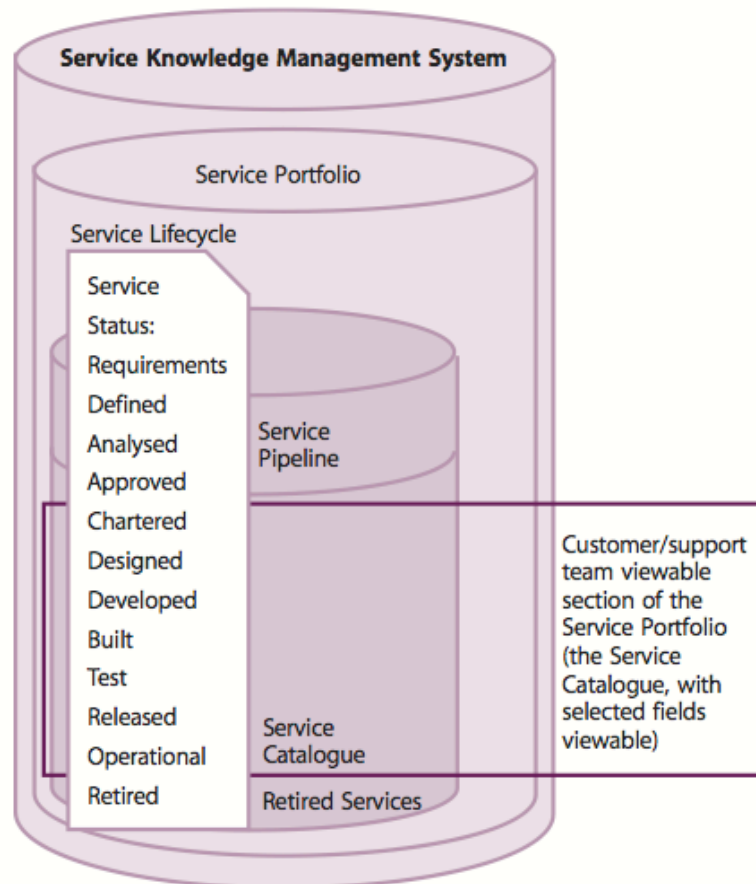


Figure 4 The Service Portfolio [OGC - Office of Government Commerce – Service Design, 2007]

The Service Portfolio is the main source of information on the requirements and services and needs to be very carefully designed to meet all the needs of all its users. The design of the Service Portfolio needs to be considered in the same way as the design of any other IT service to ensure that it meets all of these needs. [OGC - Office of Government Commerce – Service Design, 2007].

5.3 Software Quality Requirements and Evaluation

Purpose of creating Software Quality Requirements and Evaluation (SQuaRE) series of standards is to unify processes: requirements specification, measurement and evaluation. [VANICEK, PAPIK, 2012].

Explanation provided by [VANICEK, PAPIK, 2012] is directly based on the drafts of ISO/IEC 25010 and ISO/IEC 25020 standards and represents the stage of SQuaRE project on the beginning of the year 2010.

SQuaRE is designed to assist in developing and acquiring products with specification and evaluation of quality requirements. *“It establishes criteria for the specification of software product quality requirements and their evaluation. It includes a two-part quality model for aligning customer definitions of quality with characteristics of software product.”* [VANICEK, PAPIK, 2012].

“SQuaRE consists of following five divisions:

- Quality Management Division (2500n),*
- Quality Model Division (2501n),*
- Quality Measurement Division (2502n),*
- Quality Requirements Division (2503n), and*
- Quality Evaluation Division (2504n).”* [VANICEK, PAPIK, 2012].

We are going to focus on Quality Measurement Division (2502n). It consists of three different parts: quality in use, external and internal software quality.

5.3.1 Model for External and Internal Software Product Quality

Software product quality model consists of software product quality attributes divided into eight characteristics, each containing set of sub characteristics [VANICEK, PAPIK, 2012]. We can see the composition of characteristics on Figure 5.

“Today the attributes and measures typical for these characteristics are still not proposed. Examples of software product quality measure are given in ISO/ETC TR 9126-2 and ISO/IEC 9126-3 (to be replaced by ISO/IEC 25022 and ISO/IEC 25023)” [VANICEK, PAPIK, 2012]. We can look at short definitions and notes of these characteristics and sub characteristics that are provided in [VANICEK, PAPIK, 2012]. Brief summary is showed on Figure 6.

5.3.1.1 Functional Suitability

Functional suitability is “*the degree to which the product provides functions that meet stated and implied needs when the product is used under specified conditions.*” [VANICEK, PAPIK, 2012].

Functional appropriateness is “*the degree to which the set of functions is suitable for specified tasks and user objectives.*” [VANICEK, PAPIK, 2012].

Accuracy is “*the degree to which the product provides the correct results with the needed degree of precision.*” [VANICEK, PAPIK, 2012].

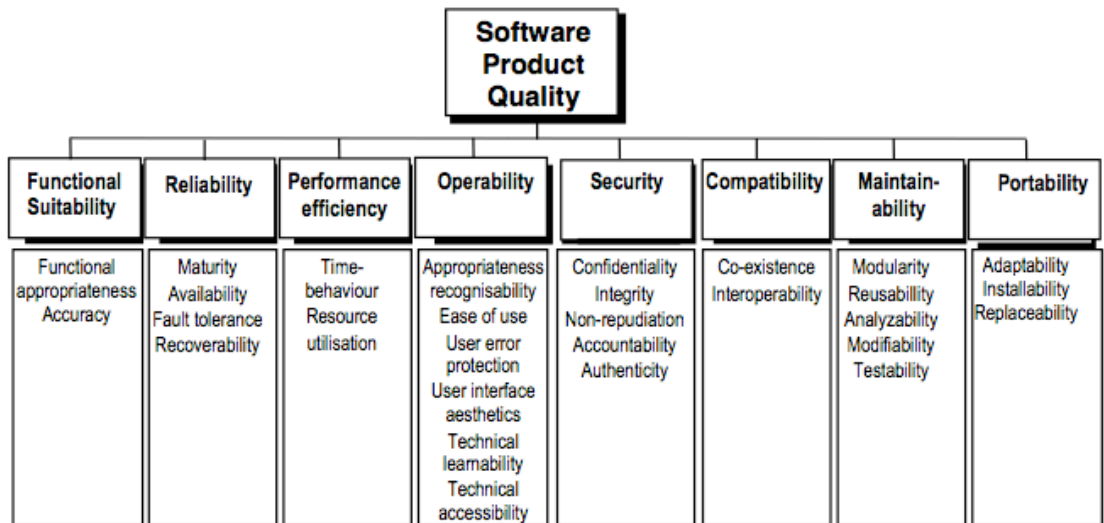


Figure 5 Software product quality model [VANICEK, PAPIK, 2012]

5.3.1.2 Reliability

Reliability is “*the degree to which a system or component performs specified functions under specified conditions.*” [VANICEK, PAPIK, 2012].

Availability is “*the degree to which a system or component is operational and accessible when required for use.*” [VANICEK, PAPIK, 2012]

Recoverability is “*the degree to which the product can recover the data directly affected and re-establish the desired state of the system in the case of an interruption or failure.*” [VANICEK, PAPIK, 2012]

5.3.1.3 Performance Efficiency

Performance efficiency is “*the performance relative to the amount of resources used under stated conditions.*” [VANICEK, PAPIK, 2012].

Time behaviour is “*the response and processing times and throughput rates of a system when performing its function, under stated conditions in relation to an established benchmark.*” [VANICEK, PAPIK, 2012].

5.3.1.4 Operability

Operability is “the degree to which the product has attributes that enable it to be understood, learned, used and attractive to the user, when used under specified conditions” [VANICEK, PAPIK, 2012].

Ease of use is “the degree to which the product has attribute that make it easy to operate and control.” [VANICEK, PAPIK, 2012].

Functional suitability	Security
Functional appropriateness	Confidentiality
Accuracy	Integrity
Reliability	Non-repudiation
Maturity	Accountability
Availability	Authenticity
Fault tolerance	Compatibility
Recoverability	Co-existence
Performance efficiency	Interoperability
Time behavior	Maintainability
Resource utilization	Modularity
Usability	Reusability
Appropriateness	Analyzability
recognisability	Modifiability
Ease of use	Testability
User error protection	Portability
User interface aesthetics	Adaptability
Learnability	Installability
Accessibility	Replaceability

Figure 6 Software product quality characteristics and sub characteristics [VANICEK, PAPIK, 2012]

User error protection is “the degree to which the system protects user against making errors.” [VANICEK, PAPIK, 2012].

User interface aesthetics is “the degree to which the user interface enables pleasing and satisfying interaction for the user.” [VANICEK, PAPIK, 2012].

Technical learnability is “the degree to which the product enables users to learn its application.” [VANICEK, PAPIK, 2012].

5.3.1.5 Security

Security is *“the degree of protection of information and data so that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them.”* [VANICEK, PAPIK, 2012].

Confidentiality is *“the degree of protection from unauthorized disclosure of data or information, whether accidental or deliberate.”* [VANICEK, PAPIK, 2012].

Accountability is *“the degree to which the actions of an entity can be traced uniquely to the entity.”* [VANICEK, PAPIK, 2012].

5.3.1.6 Compatibility

Compatibility is *“the degree to which two or more systems or components can exchange information and/or perform their required functions while sharing the same hardware or software environment”* [VANICEK, PAPIK, 2012].

Co-existence is *“the degree to which the product can co-exist with other independent software in a common environment sharing common resources without any detrimental impacts.”* [VANICEK, PAPIK, 2012].

Interoperability is *“the degree to which two or more systems or components can exchange information and use the information that has been exchanged.”* [VANICEK, PAPIK, 2012].

5.3.1.7 Maintainability

Maintainability is *“the degree of effectiveness and efficiency with which the product can be modified”* [VANICEK, PAPIK, 2012].

Modularity is *“the degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.”* [VANICEK, PAPIK, 2012].

Testability is *“the ease with which test criteria can be established for a system or component and tests can be performed to determine whether those criteria have not been met.”* [VANICEK, PAPIK, 2012].

5.3.1.8 Portability

Portability is *“the degree to which a system or component can be effectively and efficiently transferred from one hardware, software or other operational or usage environment to another.”* [VANICEK, PAPIK, 2012].

Adaptability is *“the degree to which the product can effectively and efficiently adapted for different specified hardware, software or other operational or usage environments.”* [VANICEK, PAPIK, 2012].

Installability is *“the ease with which the product can be successfully installed and/or uninstalled in a specified environment.”* [VANICEK, PAPIK, 2012].

Replaceability is *“the degree to which the product can be used in place of another specified software product for the same purpose in the same environment.”* [VANICEK, PAPIK, 2012].

5.4 Software Technologies

To be able to have actual output of designed service we need to have access to proper technologies. We are going to look closer on web application based approach.

5.4.1 Internet and Web

“The names The Internet and The Web are commonly used to refer to some part or all, of the global computer communication network with which the world is familiar. In fact they are two different entities, with fairly well defined boundaries. Generally speaking, the Web is an Internet application; that is, it depends upon Internet services to operate.” [GROVE, 2009].

“Today the Internet and the Web are governed by cooperative bodies that include governments, businesses, and individual as members.” [GROVE, 2009]. [GROVE, 2009] also mentions important groups or committees taking part in Internet:

- **The Internet Society** (ISOC), which is responsible for development of Internet architecture and protocols.
- **Internet Architecture Board** (IAB), part of ISOC. Decisions concerning Internet development.
- **Internet Engineering Task Force** (IETF), part of ISOC. Decisions concerning Internet development.
- **Internet Corporation for Assigned Names and Numbers** (ICANN) decides how domain names and Internet addresses are assigned.
- **World Wide Web Consortium** (W3C) oversees the development of web protocols.

Large number of protocols provides control over interaction and operation of Internet components. [GROOVE, 2009] mentions the most relevant of these form an application development perspective:

- Internet Protocol (IP)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

“Just as the OSI Model uses a layered architecture, the Internet Protocol Suite is also based around a layered mode” [MILLER, 2009]. Open System Interconnection (OSI) Reference model is network communication

architecture. How the Internet Protocol Suite model, which is also based around layered model, corresponds to OSI model is showed on Figure 7 and each layer is discussed below.

- **“The Network Access or Physical Layer.** Unlike the Physical and Data Link layers in the OSI Architecture, the Network Access or Physical Layer from the Internet Architecture is not defined by any specific Internet Standards. Instead the upper three layers built upon network access methods defined by standards making bodies such as the IEEE.” “As a result, this layer is broadly equivalent to the combined functionality of the OSI Physical Data Link Layers” [MILLERS, 2009].

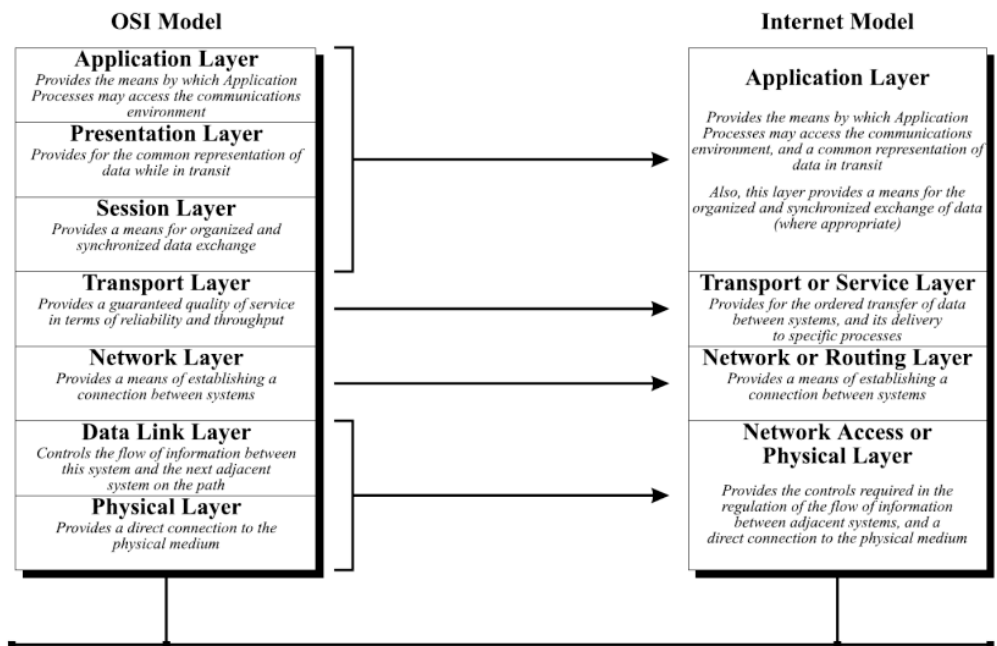


Figure 7 A comparison of OSI and the Internet Architecture models [MILLER, 2009]

- **“The Network or Routing Layer.** This layer is broadly equivalent to the Network Layer in the OSI Architecture and is responsible for the routing and delivery of data between networks. Such operations may become quite complex where differences exists between the low level naming or addressing schemes used, and this layer must therefore make the appropriate translations where required. The Internet Protocol (IP) together with the Internet Control Message Protocol (ICMP) is used at this layer, and since it must function with all network types is independent of all underlying network architectures.” [MILLER, 2009].
- **“The transport or Service Layer.** This layer is broadly equivalent to the Transport Layer in the OSI Architecture; however some

Session Layer functionality is also inherent here. In general, this layer provides a delivery to a specified Application layer port, ensuring data delivery to a specific application process.” [MILLER, 2009]

- **“The application Layer, in the OSI Architecture, provides the end-user application process with access to the communication environment.”** “The Application Layer is broadly equivalent to the Presentation and Application Layers of the OSI Model, although since certain protocols also provide synchronization and error detection, it is difficult to show clear demarcation between the layers of two model types. Certainly some would argue that the Internet Architecture is not as feature rich as that of the OSI Architecture, yet it serves the same purpose and performs the task exceptionally well” [MILLER, 2009].

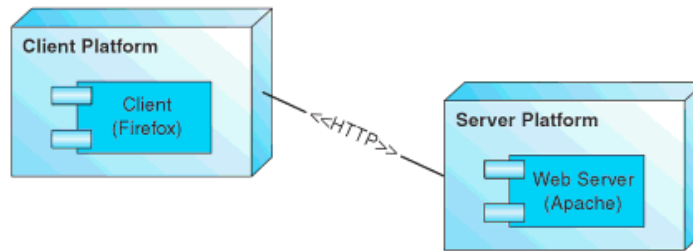


Figure 8 Client-server architecture [GROVE, 2007]

5.4.2 Web Application Environment

By general definition we can describe as a web application any application that uses a web browser as a client. Web application is usually accessed over network e.g. Internet. It uses client-server architecture. “In the client-server architecture, the client is a component that makes requests for service from another component, the server, that satisfies those requests. The client and server are connected via some type of network.” [GROVE, 2007]. We can see an example of client-server architecture on Figure 8.

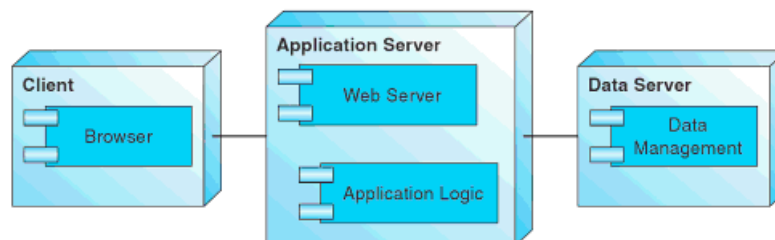


Figure 9 Three-tier architecture [GROVE, 2007]

“Client-server architectures can be categorized by the number of tiers, or layers, that they comprise. A tier is a part of an application that can be assigned to one or more similar platforms. Typical tiered components include user interface, business logic, data management, web server, and firewall. Standard architectural styles include two-tier, three-tier, and multi-tier designs” [GROVE, 2007]. We can see an example of a three-tier design on Figure 9. Three-tier design is often used in web application development. We are going to mention software that can be used in each part of three-tier architecture.

5.4.2.1 Browser

Browser is situated on the Client side of three-tier architecture. *“A **browser** is a software application for retrieving, presenting and traversing information resources on the World Wide Web. An information resource is identified by an Uniform Resource Identifier (URI) and may be a web page, image, video or other piece of content. Hyperlinks present in resources enable users easily to navigate their browsers to related resources. A web browser can also be defined as an application software or program designed to enable users to access, retrieve and view documents and other resources on the Internet.” [Web browser, 2012].* Content that is served to browser is a web page mostly in HyperText Markup Language (HTML). Version of some web browser is pre-installed in most of common operation system I think it is a good approach to use it as a user interface with any application because it reduces cross platform compatibility issues, though that not all browsers support the same standards this has to be considered in application design. We can have a look on overview of browser statistics from October 2012 in Table 1.

Browser	Usage
Internet Explorer	16.1%
Firefox	31.8%
Chrome	44.9%
Safari	4.3%
Opera	2.0%

Table 1 Browser characteristics October 2012 [Browser Statistics]

5.4.2.2 Web Server

Web server is part of Application Server in the three-tier architecture. As they state in [YEAGER, MCGRATH, 1996] web server can be made of three things: platform, software, and information. We can see the platform as computers hardware, network or Internet connection and operating

system. Platform provides the web server space and resources to run and communicate on network. The role of software part of web server is to provide bridge between platform and information. **“Web server software receives request from Web browsers for documents over the network, deciphers each request to determine which file is needed, finds that file if it is available, and sends that file back to the Web browser over the network connection.”** [YEAGER, MCGRATH, 1996]. By Web server information we can understand any kind of document e.g. Application Logic in Application Server part in the three-tier architecture that will provide desired content.

An example of web server is WEBrick. “WEBrick is an HTTP server toolkit that can be configured as an HTTPS server, a proxy server, and a virtual-host server. WEBrick features complete logging of both server operations and HTTP access. WEBrick supports both basic and digest authentication in addition to algorithms not in RFC 2617.” [WEBrick].

5.4.2.3 Application Logic

Application Logic is a part of Application Server in a three-tier architecture. It provides information to Web Server. We can imagine a Web application framework on this position. *“A web application framework is a software that is designed to support the development of dynamic websites, web application and web services. The framework aims to alleviate the overhead associated with common activities performed in Web development. For example, many frameworks provide libraries for database access, templating frameworks and session management, and they often promote code reuse.”* [Web application framework].

Web application can use a model-view-controller architecture (MVC). *“Model-view-Controller (MVC) is a well-known design pattern that was first formalized part of the Smalltalk programming language.”* [PARSONS, 2008]. *“The MVC architecture explicitly separates the roles of the model (the underlying data), the view (what is presented to the user) and the controller (the handling component that manages user interaction and triggers appropriate updates to the model).”* [PARSONS, 2008]. Integration of MVC in three-tier architecture is displayed in Figure 10, that I prepared. Routes Dispatcher part of Application Logic represents its internal processes of navigation within the application logic.

An example of MVC open source framework is Ruby on Rails. It is based on programming language Ruby.

Ruby is a object-oriented language: “Virtually everything in Ruby is an object, and virtually all of those objects inherit a basic set of methods from the Object class. In a very real way, the Object class is the glue that binds Ruby together and lends the language its simple glance.” [OLSEN, 2011]. Nice example is how Ruby interprets: “Feed it a string and it will tell you that you have a *String*. Feed it a number and you might see that you have an instance of *Fixnum* or *Float*. Feed it *nil* and it will tell you that you have an instance of *NilClass*.” [OLSEN, 2011].

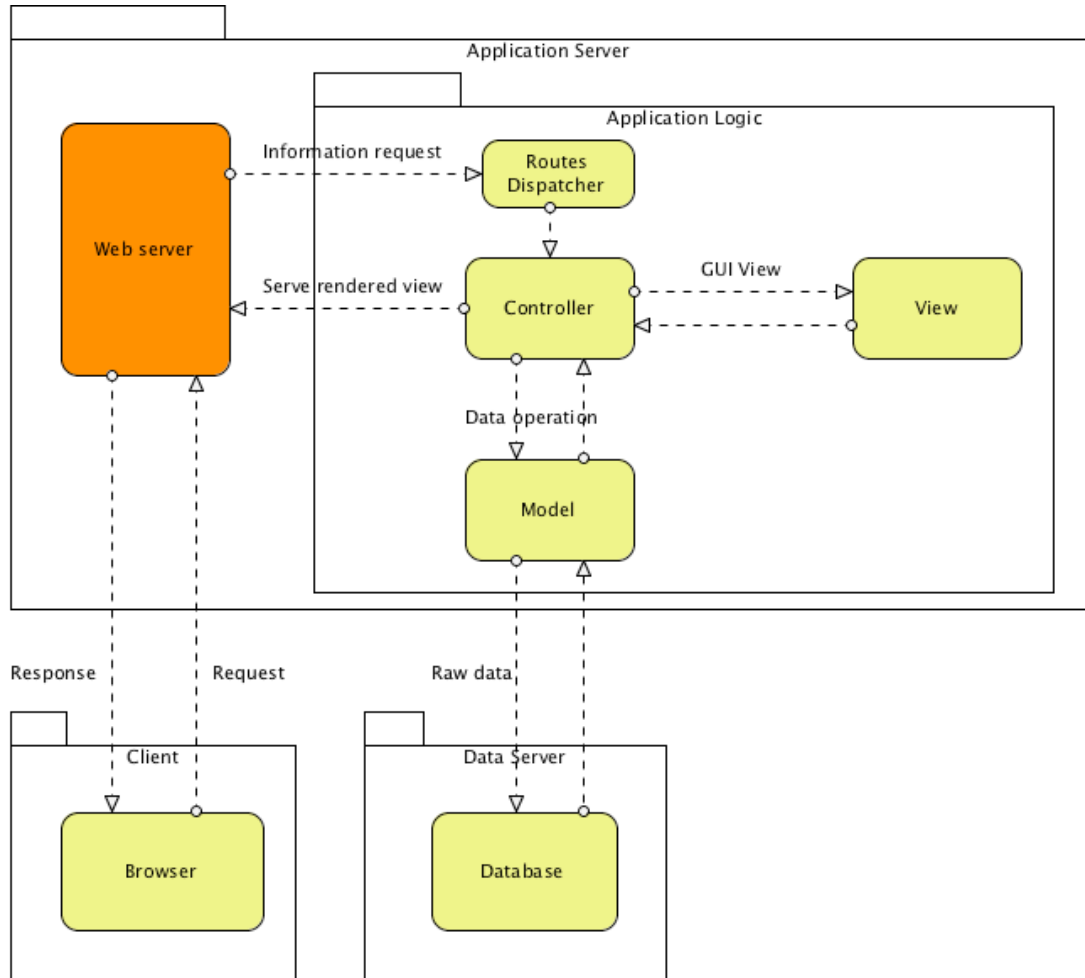


Figure 10 Three-tier architecture with MVC

As they mention in [THOMAS, FOWLER, 2009] Ruby on Rails consists from separated packages: ActiveRecord, ActiveSupport, ActionPack, ActionMailer. Interesting package is ActiveRecord it is a module in Ruby on Rails that provides access to the database. It is an object-relational mapping system that maps data from database into object instance inside Ruby on Rails framework.

Ruby has a large community of developers that helps to expand it. As Yukihiro “Matz” Matsumoto, creator of Ruby language, says in foreword in [FULTON, 2002]: “As always, I’d like to express my very greatest appreciation to the people in the Ruby community. They are the heart of Ruby’s success.”. There is a large community around Ruby on Rails framework as well. They use RubyGems package manager to distribute Ruby programs [RubyGems.org].

5.4.2.4 Data management

The third-tier part of client-server architecture host data server, which is typically represented by standard database. I found a definition of a database concept: “Database is a collection of data. It contains information about one particular enterprise. A database management system (DBMS) consists of a collection of interrelated data and a set of programs to access that data.” [KEDAR, 2009]. Different types of data models with description are listed in [KEDAR, 2009]:

- Entity relationship model (E-R) consists of a collection of basic objects, entities, and relationships among them.
- Relational model consists of data and relationship among data by a collection of tables, with number of columns with unique names. It achieves data and structural independence making the database design, maintenance, administration and usage easier than the other models.
- Hierarchical model links records together in a tree data structure. Each record type has only one owner.
- Network model is based on directed graph theory. It replaces hierarchical tree with a graph thus allowing more general connections among nodes, it is able to handle many-to-many (m:n) relationships.
- Object oriented model is based on a collection of objects. Each object, instance, have data stored in instance variables. Objects are grouped into classes.
- Object relational model combines infrastructure of object model and set of relational extenders.

An example of a database system is MySQL. MySQL is a open source (under GNU General Public license) relational database management system, that runs on more than 20 platforms [MySQL]. It is often use in web applications.

5.4.3 Software Development Environment

By software development environment I mean a set of programming tools to create a software product, an application. For this purpose exist many solutions. I briefly mention only the ones that were used during work on this thesis.

5.4.3.1 Vim for Mac OS X

Vim (Vi IMproved) is a highly configurable text editor, freely available for many different platforms [Vim]. Adding FuzzyFinder script utility [FuzzyFinder] makes Vim editor powerful and easy to use tool to handle programming code. FuzzyFinder allows Vim subdirectory search. Additional settings for Vim are done in `.vimrc` and `.gvimrc` files.

5.4.3.2 Git

To enable team of developers to handle shared programming code it is necessary to use some versioning system. Great solution is to use Git [Git]. "*Git is a free open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency*" [Git]. One solution to deliver versions of project to everybody is to set-up working repository that is available through network connection, which is usually secured by using Secure Shell (SSH) connection. Git is capable of creating branches of repository. That enables division of programing code, e.g. on production and development part. When it is desired one branch can be merged with another one e.g. development version can be merged to production.

Repository management is natively done through shell of Operating System (OS). In Mac OS X terminal we can extend usage of Z Shell by installing plugin oh-my-zsh [oh-my-zsh], which includes various plugins and plugin git, is among them. It facilitates navigation in the project by displaying current branch of git repository.

GitX is a Graphical User Interface (GUI) of Git for Mac OS X [GitX]. GitX displays list of Git repository branches and their commits. I think it is good for creating commits and revision of state of the repository but its functionality is limited and some issues have to be resolved in terminal.

5.4.3.3 Web Development Tools

As GUI of web application is displayed in browser as a HTML and it is styled using Cascading Style Sheets (CSS) we need some efficient tool to design and modify its content. Powerful tool is Firefox browser [Mozilla Firefox Web Browser] plugin Firebug [Firebug]. Useful features of Firebug are inspecting HTML and modifying style in real-time and using JavaScript console.

II. Project

6 Invoicing System

In this chapter I am going to introduce invoicing system, analyse it and evaluate its quality as a software product.

6.1 Introduction

Invoicing system with which I am going to work with in this thesis is called NuIS [NuIS]. NuIS is an online invoicing system owned and developed by Czech company GravaStar s.r.o., the provider. It is on the market since year 2009.

NuIS is targeted on small companies or sole traders that are not obligated to keep accounts e.g. using services of an external accounting company, but they issue invoices by them selves and they need to have instant access to them. Application provides issuing invoices in Portable Document Format (PDF), keep track of incomes and expenses and get statistics as a basis for further decisions.

NuIS was created because GravaStar needed to get overview of turnover, receivables, and statistics. At that time there was only one product dealing with online invoicing on the market so GravaStar decided to develop own system and provide it to other companies.

NuIS is a phonetic abbreviation for New Information System.

6.2 System Description

NuIS, the system, is a web application that is based on three-tier architecture. Users can access it any time through Internet. Main feature is invoice management system, which includes keeping track about users company and customers, divisions, related to invoicing. Invoices in status paid are also kept as incomes in incomes component of the system. Supplement to this

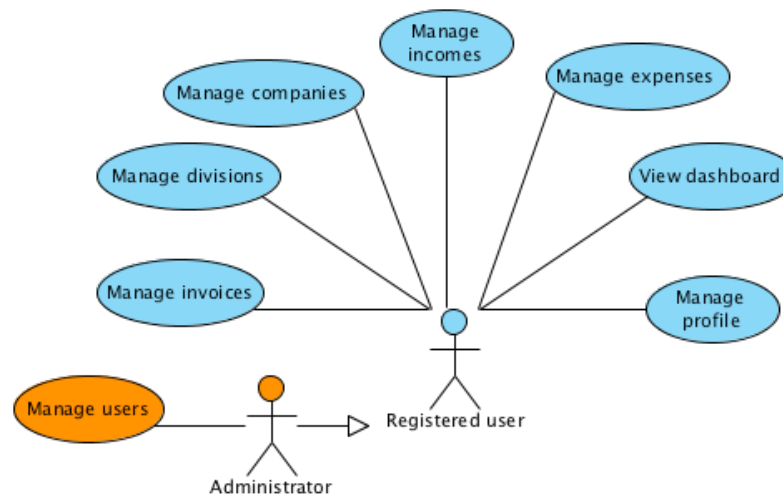


Figure 11 Basic features of NuIS

component is expenses component that enables tracking company's expenses. Information about users mother company and user data are in user profile section. Dashboard component displays summary statistics and information for user. Access to system is secured by login name and password. User management is accessible through administrator account. Main features of the system are shown on Figure 11.

System offers three price plans for user accounts:

- **Free** – up to three invoices per month;
- **Optimum** – up to ten invoices per month;
- **Profi** – unlimited invoices per month.

Free plan is without charges. Optimum and Profi are prepaid for one year according to pricelist. User has possibility to upgrade to higher plan any time.

To illustrate overview of systems internal structure in more I created UML (Unified Modeling Language) diagram showing all system's models and their relations, diagram showed is in Figure 18. Different colors help to distinguish model classification to system sections. For example models involved in invoicing are marked with yellow color: Company, Invoice, Division, Currency, Item.

6.2.1 Invoices Management

Invoice management contains most of the business logic of the system and it is directly connected to other components. To be able to issue invoice user, registered and logged in the system, has to fulfil following requirements:

- Set-up mother company. Mother company represents users own company in the system. User can have only one mother company.
- Set-up division. Division represents department of company in the invoicing system. At least one division has to be created.

While issuing new invoice specified attributes have to be filled in these attributes are defined by legislation of Czech republic. For every invoice system requires purchasing company record, which can be selected from list (using suggest feature) or directly added in new invoice form. System automatically generates invoice number respecting format YYDDNNNN. YY is two-digit representation of current year DD is two-digit representation of selected division and NNNN is invoice serial number for current year and selected division. One or more items have to be added to invoice. Item represents product or service that is billed to purchasing company.

From user point of view invoice has two main states: generated and paid. By marking invoice as paid user notifies that the payment for invoice was covered. User can retrieve list of unpaid invoices or invoices that are delayed, according to parameter collectable on. Once invoice is marked as paid it cannot be destroyed or edited. To remove invoice from accounts

it is necessary to issue not credit. To notify receiving cash invoice simple receipt can be issued for every invoice. Invoice, note credit, and simple receipt are downloaded in PDF format. System provides possibility to attach invoice PDF to an email that can be send to several email addresses e.g. purchasing company, user's company accountant.

6.2.2 Incomes Management

This section of system is designed for keeping track of user's company incomes. System has two kinds of incomes: invoice and without invoice. When an invoice is marked as paid the system creates new income. Income without invoice includes only name, price, whether Value Added Tax (VAT) was included and if it was paid. It is possible to register planned incomes without invoice. This section is not included in statistics.

6.2.3 Expenses Management

This section of system is designed for keeping track of user's company expenses. Expense can be planned for certain day. Recurrence can be assigned if it is going to be repeated. System keeps track if it is paid or not.

6.2.4 Dashboard

This is the section where user is redirected after logging in to the system. User can see statistics based on invoices and expenses. User can see summary for last month, quarter, year, and list of three upcoming expenses and unpaid invoices.

6.2.5 Companies Management

This section is designed to serve as an address book of purchasing companies. Mandatory and additional (e.g. email address, phone) information about company is recorded. If a company has any invoices assigned it is possible to display its invoices list.

6.2.6 Divisions Management

In this section user can manage its company's divisions. At least one division has to be set-up.

6.2.7 Profile Management

In this section user can manage information about him including price plan, if company is payer of VAT, system language, and some pre-set attributes.

System provides possibility to invite a user. This feature enables creating team of people, which can share same mother company. User register new account and waits for invitation from user that already uses the system.

6.3 SWOT Analysis

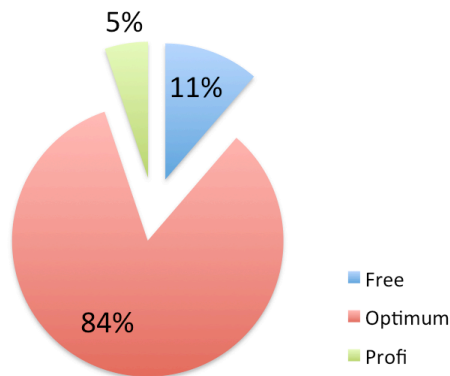
In this chapter the system is described from business point of view. Results from SWOT analysis are going to support further decisions in development or changes of main areas of research. External and internal factors are examined.

6.3.1 External Factors

6.3.1.1 Customers

System's customers, users, are small companies or entrepreneurs. I had access to system production database containing data until first quarter of year 2011. At that time system had hundreds of users. Out of total number of users 22% issued at least one invoice. To differentiate from user who just tested the system was selected group of users having issued more than 5 invoices, active users. Out of total number of users 7% were active users.

Distribution of price plans



Issued at least one invoice

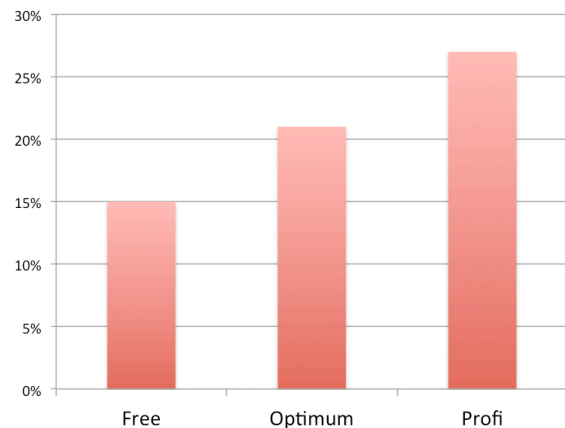


Figure 12 Usage of system

Distribution of price plans can be found in Figure 12. This graph shows how many users decided to take certain price plan. Good result is that most of users (84% Optimum, 5% Profi) decided to prepay optimum price plan indicating that price plan is set well. Customers do not have problem to pay for the service. In right part of Figure 12 is displayed chart Issued at least one invoice saying how many of users from each plan issued at least one invoice. An average of 20% users used the system to issue at least one invoice even if most of users had prepaid programme.

Communication with customers is done through individual or mass email, and notifications on dashboard after login in to the system. The provider asks for customer's feedback when new feature is released or when significant change was made in the system. To get connected with customers and potential customers NuIS is presented on social network Facebook¹ where important messages and announcements are placed.

6.3.1.2 Technology

The system is using modern web technology. Combination that it is based on object framework Ruby on Rails and that its design is well thought through give it an advantage. It is easy to develop or redesign its components. System GUI was redesigned to provide better user experience and to be able to compete with competition. Emerging technology and trends in web application GUI are more advanced every year it is important to observe the situation.

6.3.1.3 Competitors

There are many same or similar products on the online invoicing market. There were not as many competitors as today when the system was deployed on the market. Competitors offer similar product, or invoicing management is extending offered service (e.g. whole accounting system). Some features are missing in the system e.g. API to interact with other customer's systems.

Competitors do offer various services; so-called demo feature is widely used. It offers user to try out system without actual registration. Another example of such service is issuing invoice without registration and leaving any data at the system.

Result of query in Google search engine² (not logged in and as location chosen Czech republic): "fakturace online" (online invoicing in Czech language) displays reference to the system on the first page at last position. Result of the same query in Seznam search engine³ did not returned any reference to the system.

¹ Facebook [online], [16.11.2012], Facebook, <http://www.facebook.com/>

² Google [online], [16.11.2012], Google, <http://www.google.cz>

² Google [online], [16.11.2012], Google, <http://www.google.cz>

³ Seznam [online], [16.11.2012], Seznam, <http://www.seznam.cz>

6.3.1.4 Suppliers

The system's availability and performance depends on its application-hosting supplier. So far the company that provides hosting to the system caused no downtime.

6.3.2 Internal Factors

6.3.2.1 Current Performance

Having hundreds of users is still far before exceeding system capacity limitations, fluent flow, of current architecture so system there is plenty of space for new users of the system.

Error accrument it is automatically reported to developer team and is evaluated. If error is caused by unavailability of key part or function of the system it is immediately fixed.

6.3.2.2 Employee Skills

People in development team are involved in all projects of GravaStar's projects. As GravaStar's main area of business is web application development learning new technologies and gaining new skills in this area is necessary part of their job. Knowledge of most technologies is passed to every team member therefore in case when employee is unwell it is easier to take over his responsibilities.

6.3.2.3 Technical Mastery

The system's business logic can be copied from competitors as same as its development can be inspired by competitors, their features or mistakes. What is not so easy to copy is logic inside the application. Well-designed application architecture can be advantage over competition.

Approach to graphical design of the application is also each company's specific feature. Using knowledge from web application development combined with creative graphic is advantage that not many competitors have.

6.3.2.4 Company Culture

As a company with flat organizational structure and less than 10 employees it people know each other very well. Friendly atmosphere creates environment where personal and collective goals are in harmony. This helps with completing complex task as employees support each other and the company's decisions.

6.3.3 Evaluation of Internal and External Factors

Summary of external and internal factors is interpreted by defining strengths, weaknesses, opportunities, and threats, is displayed in Table 2.

<p>Strengths</p> <ul style="list-style-type: none"> • System capacity • Application architecture • Knowledge of technologies • Creative design 	<p>Weaknesses</p> <ul style="list-style-type: none"> • Open business logic • Small team of developers • Error handling
<p>Opportunities</p> <ul style="list-style-type: none"> • Ease of development • Paying customers • Communication channels 	<p>Threats</p> <ul style="list-style-type: none"> • Big competition • Long term customers • Supplier reliability

Table 2 SWOT summary

Listing and evaluating core competencies is displayed in Table 3. Evaluation is based on scale 5 = very strong; 1 = very weak. System is not compared to any specific product. Marks are correlated between comparison to competition in general and internal evaluation.

	NuIS
Competencies	
Time on market	5
Product quality	3
Satisfaction of customers	2
Developing and attracting human talent	3
Flexible development	4
Project management skills	2
IT systems	3
Critical assests	
Brand power	3
Physical plant	3
Strategic partners	1
Distribution networks	2

5 = very strong; 1 = very weak

Table 3 Core competencies and resources

Based on this analysis new strategy can be formulated. It focuses mainly on improvement and development in these areas:

- Investigate why users, customers, decide not using system even after choosing prepaid programme. Including
- Make changes based on result of suggested investigation to satisfy users requirements.
- Explore possibilities of joining partnership with another company or possibilities how to promote system on the market.

6.4 Software Product Quality Evaluation

This chapter provides detailed information about product's quality. I am going to follow Software product quality model, which is described in 5.3 Software Quality Requirements and Evaluation chapter.

6.4.1 Functional Suitability

In this section we will focus on how the product fits its purpose, provides the right functions that are expected from it.

6.4.1.1 Functional Appropriateness

This product is designed to allow its user to manage invoices online through common web browser. For this purpose it contains modules user, invoices and companies. It allows to show or download invoices in PDF format or to send them via e-mail to desired address. Product does not cooperate with any other application it does not allow to export data in any form. There is no possibility to connect it to some other accounting software. If we count this as an implied need we can say it is 80% suitable for this task.

6.4.1.2 Accuracy

If correct data are provided it provides exact results. As system is counting company finances it cannot make mistakes.

6.4.2 Reliability

Product reliability is going to be described by analysing production log of the system. Data gathered from this log describes system as whole. User individual usage is omitted instead data generated by all users are taken in account. Doing this provides general overview how the system works.

6.4.2.1 Availability

Data in our production log contains records of user requests from 2009-06-26 13:16:01 to 2012-03-08 08:52:08 which is basically the time that product is available. Total time analysed is 986 days and total number of analysed request 158072.

Number of failed requests is 6058. The failure rate is 3.8% (this number represents all errors that were captured from the start of production, 93% were uncaught routing errors). This does not mean that the system was down, but only particular action (sometimes under specific circumstances) was unavailable, until it was repaired.

6.4.2.2 Recoverability

To re-establish desired state depends on what and how significant error has occurred and who is responsible to make the fix. If it is the case of error in the core program of the application it can be fixed (or hot-fixed) within 1 – 2 hours.

6.4.3 Performance Efficiency

As user hardware requirements are low, application runs on server, it is enough to have sufficient computing power to browse HTML webpages, run JavaScript and be able to open PDF files. User must have Internet connection (recommended speed 1 Mbit/s).

6.4.3.1 Time Behaviour

Time behaviour is affected by many factors. One is how many users are requesting the system. System can serve only three requests at once. If there are more than three requests at a time others have to wait in a FIFO (First In, First Out) queue. Another factor affecting time behaviour is what action is user requesting. Statistics on request duration is displayed on output

Request duration - by mean	Hits	Sum	Mean	StdDev	Min	Max	95 %tile
InvoicesController#old_note_credit.pdf [GET]	1	4.80s	4.80s	0.00s	4.80s	4.80s	4.74s-4.90s
UsersController#list.html [GET]	602	39m60s	3.99s	2.73s	0.00s	19.89s	0.25s-13.52s
DashboardController#update_year_statistics.html [POST]	1334	1h07m29s	3.84s	4.62s	0.00s	26.32s	0.23s-18.37s
PricePlansController#order_price_plan.html [GET]	6	14.36s	2.39s	4.13s	0.25s	10.60s	0.24s-10.62s
OpenInvoicesController#division_company_ares.html [POST]	1	2.29s	2.29s	0.00s	2.29s	2.29s	2.26s-2.33s
WizardController#company_ares.html [POST]	193	7m08s	2.22s	1.81s	0.04s	21.03s	0.05s-6.55s
CompaniesController#from_ares.html [POST]	607	21m52s	2.16s	1.05s	1.67s	11.95s	1.72s-6.76s
CompaniesController#from_ares_ajax.html [POST]	98	3m24s	2.08s	0.53s	0.05s	4.59s	1.58s-3.85s
DashboardController#move_quarter_one.html [POST]	16	30.57s	1.91s	1.43s	0.16s	3.45s	0.16s-3.49s
OpenInvoicesController#company_ares.html [POST]	1	1.85s	1.85s	0.00s	1.85s	1.85s	1.83s-1.89s
DashboardController#update_quarter_statistics.html [POST]	1364	37m09s	1.63s	3.80s	0.00s	27.64s	0.13s-16.94s
DashboardController#update_month_statistics.html [POST]	1365	35m35s	1.56s	3.53s	0.00s	26.57s	0.13s-16.94s
DashboardController#move_year_two.html [POST]	20	28.49s	1.42s	2.04s	0.12s	7.03s	0.12s-7.18s
InvoicesController#show.pdf [GET]	4555	1h45m34s	1.39s	3.03s	0.00s	1m06s	0.01s-10.62s
UsersController#create.html [POST]	578	11m07s	1.15s	1.43s	0.04s	11.38s	0.04s-6.34s
DashboardController#move_quarter_two.html [POST]	13	13.38s	1.03s	0.76s	0.17s	2.27s	0.16s-2.30s
InvoicesController#note_credit.pdf [GET]	25	23.82s	0.95s	1.03s	0.03s	4.80s	0.03s-4.82s
InvoicesController#send_off.html [POST]	78	1m07s	0.86s	1.74s	0.02s	10.40s	0.02s-7.94s
DashboardController#index.html [GET]	6479	1h17m00s	0.71s	1.61s	0.00s	19.16s	0.00s-7.21s
CompaniesController#index.html [GET]	4212	49m18s	0.70s	0.96s	0.00s	7.08s	0.04s-3.66s

Figure 13 Request duration - by mean

of Request-log-analyzer⁴ on Figure 13.

6.4.4 Operability

6.4.4.1 Ease of Use

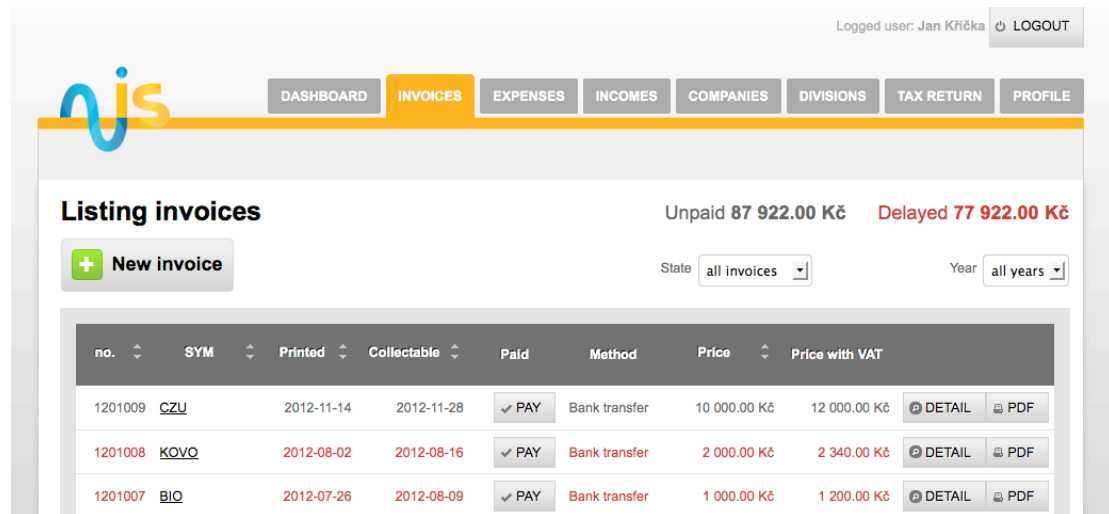
Product environment was designed regarding ergonomic, visible and navigable aspects of use. It helps user to follow the workflow intuitively.

6.4.4.2 User Error Protection

System checks user's entry data by validating input parameters. It checks required fields and their data types. First level of protection is on the client side in users browser (JavaScript must be enabled) and second level is on server side by validating models parameters (user cannot influence that).

6.4.4.3 User Interface Aesthetics

I find user interface designed to look like a modern application that is attractive and can compete between other products example of interface can be seen on Figure 14 (Example screenshot was taken after changes related to this thesis were implemented, GUI design remained the same.).



Logged user: Jan Kříčka LOGOUT

DASHBOARD INVOICES EXPENSES INCOMES COMPANIES DIVISIONS TAX RETURN PROFILE

Listing invoices Unpaid 87 922.00 Kč Delayed 77 922.00 Kč

+ New invoice State: all invoices Year: all years

no.	SYM	Printed	Collectable	Paid	Method	Price	Price with VAT		
1201009	CZU	2012-11-14	2012-11-28	✓ PAY	Bank transfer	10 000.00 Kč	12 000.00 Kč	DETAIL	PDF
1201008	KOVO	2012-08-02	2012-08-16	✓ PAY	Bank transfer	2 000.00 Kč	2 340.00 Kč	DETAIL	PDF
1201007	BIO	2012-07-26	2012-08-09	✓ PAY	Bank transfer	1 000.00 Kč	1 200.00 Kč	DETAIL	PDF

Figure 14 Example of NuS GUI

⁴ Request-log-analyzer – is a command line tool that analyzes request logfiles; wvanbergen/request-log-analyzer [online], [18.11.2012], GitHub, <https://github.com/wvanbergen/request-log-analyzer>

6.4.4.4 Technical Learnability

Interface is very intuitive it guides user through every section. To important fields are attached tooltips with explanation. If user has elementary knowledge of invoicing it is easy to use.

6.4.5 Security

6.4.5.1 Confidentiality

System protects user data by separating user accounts. Every user has unique login name and password that is kept hashed in database (md5 encryption) and checked during logging in. System is checking users presence, and rights before every requested action.

6.4.5.2 Accountability

Every record has specific user to which he has access (or group of connected users) and can manipulate with it. The system does not keep track of changes as log. Some changes can be traced from system console interface. Important models such as invoice model are not removed from database but marked as deleted.

6.4.6 Compatibility

6.4.6.1 Co-existence

System can co-exist with other applications in the same environment. Only one user logged in to the system from one browser application on client side is possible.

6.4.6.2 Interoperability

System cannot exchange information with other applications. It does not have export or import function. Data can be displayed in the application (invoices can be downloaded or send as attachment in PDF file).

6.4.7 Maintainability

6.4.7.1 Modularity

System is based on models that contain business logic. Models or their instances can interact together. Adding new feature or module is simple. Create completely new model and add behaviour to existing models if necessary. Have to check backward compatibility when changing behaviour of old model.

6.4.7.2 Testability

There are several types of test. Testing models (logic and calculations), performing actions (preparing data) and generating views (check that data are displayed correctly). When some change is done in model behaviour model tests are going to reveal if it affected other system functions.

6.4.8 Portability

6.4.8.1 Adaptability

There are low hardware requirements. Operating system must support installation of common web browser and Internet connection.

Operating system compatibility was tested on operating systems MS Windows (XP, 7), Mac OS (10.6, 10.7). Web browser compatibility was tested on browser MS IE (7 and higher) on MS Windows, Mozilla Firefox (5 and higher) on Mac OS, Safari (5) on Mac OS, Google Chrome on Mac OS. In all browsers the application is showed correctly except MS IE, which has some difficulties processing CSS standard.

6.4.8.2 Installability

This application does not need any installation. Its prerequisite is installed version of web browser in operating system of a computer (or any device capable displaying webpages). Application itself does not require any disk space for installation.

6.4.8.3 Replaceability

Some other product can replace this system. There are many similar products available on the market. Absence of feature exporting data from the system disables possibility to migrate data to another application. Users only option would be transfer all data manually.

6.4.9 Software Product Quality Summary

The system's main benefits are accessibility: as web application; modularity: can be easily modified, which can support make further development and expanding functionality decisions; easy to redesign: can keep modern and ergonomic design. Weakness can be represented by absence of importing/exporting feature, not functioning without Internet connection, overloading of server with increasing number of users. Considering previous analysis system can pass as online invoicing system that is suitable for commerce usage.

6.5 Research in Main Areas

After system analysis and description we came to discuss given main areas of research. We formulated these topics based on analysis of invoicing system with GravaStars's development team combining company's executive director's project vision. In following chapters I am going to provide information about each area's state, discuss its problematic, and suggest or implement my solution.

During design and implementation process main areas of researched were approached as system services. I find ITIL framework (introduced in 5.2) business approach of designing services very effective and structured. As it is designed for large project development, involving many project roles, I decided not to apply this framework in this work. Though I was inspired by some of its practices in design process. Balance of design was one of them. My time spend on this work was not valued by money but still I tried to keep resource, functionality, and schedule in balance as I would expect in commercial development. Business requirements were the basis of solution design to provide user the best possible service.

6.5.1 Registration Process

After going through initial information provided at system's homepage the registration process is user's first interaction with the system. This first experience can influence user's opinion on the whole system. It has to give the impression of well-organized and easy to use system. Intuitive approach of taking user through the registration process ended by user being able to issue invoice was considered essential. The registration process state was not meeting this requirement.

6.5.1.1 Initial State

The registration process is designed to set up user account in the system. It collects general information about user:

- Selected price plan
- User's name
- Login name
- E-mail address
- Password

From these attributes only user's name is going to be displayed on invoice. Other information about user's company has to be entered in other part of the system.

To be able to issue first invoice following steps are required. After successful registration user is redirected to page where he can decide create new mother company, or wait for invitation, or use the main navigation menu.

Invite user can only user who already set-up mother company and is able to issue invoices. User, which is not waiting for invitation, can proceed to creating mother company. When new mother company is created list of all companies is displayed. User still cannot issue invoice because division is not created.

Diagram showing users path from registration to new invoice is displayed on Figure 19. We can see that user is forced two times to fill another form and return back before it is possible to proceed issuing invoice.

6.5.1.2 Solution Design

Convenient would be to join all task necessary for issuing invoice in one. User after creating account, registering, would be able to issue invoice. I propose idea of two-step register form. First step would cover user's credentials and second would be focused on creating mother company. After going successfully through these two steps user will be redirected to page notifying successful registration and menu including issuing new invoice will be displayed.

This modification of registration process into register wizard should make usage of the system more intuitive and increase number of active users. Previous, unclear, process could be one of factors that such small number of users issued at least one invoice as mentioned in 6.3.1.1 Customers.

Diagram showing proposed registration process workflow is displayed on Figure 20.

I would recommend designing a new controller controlling the whole registration process. Now various controllers maintain these actions: UsersController, CompaniesController, and DivisionsController.

6.5.1.3 Soution Implementation

To implement proposed registering process I decided to create new controller WizzardController performing related actions. Two main forms were designed to gather important data: user form and user's company form. User related actions are: user, create_user. Company related actions are: company, company_ares, and create_company. List of created actions important for registration process is in Table 4.

Action	Description
user	Renders user form. Prepares new user. Assigns price plan according to param[:programme], entry record in routes.rb: map.signup '/signup/:programme', :controller => 'wizard', :action => 'user'
create_user	Creates, saves (if valid), user that is passed from action user. Handles creating selected price plan and validate coupon if present. Redirects to company or invited user, or renders new user form.
company	Renders form for search company on ARES ⁵ system. Prepares new company.
company_ares	If found, through VIN, prefills company data according to ARES system. Called as AJAX request from company form. Renders partial with new company form, or ARES search form.
create_company	Creates, saves (if valid), company from new company form. To saved company generates division. Redirects to welcome, or renders new company form.
welcome	Renders confirmation message and menu.

Table 4 Registering process actions

Implementation of company_ares action handling company data prefill:

```

def company_ares
  begin
    @company = Company.prepare_from_ares(:ico =>
params[:vin])
    @notice = I18n::t('not_found_on_ares') unless
@company.vin
  rescue
    @company = Company.new
    @notice = I18n::t('ares not working') unless
@company.vin
    @manual = true
  end
  @company.email = current_user.email

```

⁵ ARES - Access to Registers of Economic Subjects registered in the Czech republic, ARES [online], [24.11.2012], <http://www.info.mfcr.cz/ares/ares.html.en>

```
render :partial => "company_ares"  
end
```

Implementation of registration process forms is showed: user part on Figure 21, check on ARES company part on Figure 22, and company form in company part on Figure 23.

6.5.2 Free Invoicing

Free invoicing is a new module that is going to be attached to the system. It will allow users to issue invoices without registration. From the view of the system two groups of users are going to co-exist: registered and not registered.

This module's purpose is to provide easier access to the system and provide its experience to wider group of people. Showing how the system looks and how smoothly it works should attract more potential users, as the barrier of registration will disappear. It also a response to situation on competition on the market described in 6.3.1.3 Competitors.

System will provide similar form for creating invoice, as the registered version, except it will have section for entering user's company data. Both companies will be possible to pre-fill through entering company's VIN⁶ from ARES system.

System will provide possibility to view and edit invoice before serving PDF file. Open invoices are going to be kept in system only as PDF files. Record in DB for every open invoice is going to contain issuing company vin, division, total price, and timestamp.

6.5.2.1 Solution Design

As there are many steps between filling new open invoice form and serving the PDF file including storing appears a question where to store open invoice data: If is better to create a new model inheriting regular invoice model or use existing one for regular invoice. As the model for regular invoice is validating many attributes and relations it would be difficult to design model that inherits from it, and it would include lot of code refactoring.

After analysing options it seems to be more efficient (time and code saving) to use existing invoice model without saving it to DB. Therefore there would no longer be problem validating the model. Because of this approach I revealed large data integrity issue in the system.

⁶ VIN – system's abbreviation for Company Identification Number, Czech IČO

As system is model based it saved data only through relations between objects, model instances. For example data about mother company are not saved in invoice object but accessed from mother company object. If any information would be changed it would reflect to all previous invoices, which is against business logic and may cause some legal to the user issues as well.

To fix data integrity and prepare for open invoice solution it would be necessary to extend invoice model with attributes from company (both mother company and issued to company), division and user models and fill them before saving regular invoice.

For fixing data integrity all actions especially views generating invoice outputs have to be refactored to work with new attributes instead of relations (actually DB queries represented in objects). Implement new method freezing attributes before saving invoice.

To preserve open invoice across multiple requesting different actions I decided to store it to session variable, which solves issues with saving invoice to DB.

Separate forms, view and PDF template have to be made.

6.5.2.2 Solution Implementation

I prepared invoice model to hold necessary data, and to fixed the integrity bug. At the beginning analysis of required attributes was done, which led to creation of migration `AddFreezeAttributesToInvoice`, its content is in attached file 10.1 (and all required attributes can be found there). `Freeze_attributes` method was added to invoice model and called before saving invoice. Refactoring of code depending on this change was done, resulting in fixing the data integrity issue.

Storing whole invoice in session required DB management of sessions. Ruby on Rails natively provides this feature. It is necessary to create table sessions with specific attributes and configure the `environment.rb` by adding:

```
config.action_controller.session_store = :active_record_store
```

`OpenInvoicesController` was created including actions handling open invoicing workflow; main methods are described in Table 5.

Action	Description
new	Renders new open invoice form. Nulls session[:open_invoice], prepares new invoice with three items.
prepare	Checks invoice validity and perform calculations. Renders show or redirects to edit action. Saves invoice to session[:open_invoice]
edit	If invoice is in session renders edit form or redirects to action new.
show	Generates PDF file and free_invoice_stamp. Redirects to action final.
final	Renders link to download invoice.
serve	Send demanded file.
company_ares	AJAX action prefilling user's company data from ARES.
division_company_ares	AJAX action prefilling user's customer company data from ARES.

Table 5 Actions in OpenInvoiceController

Invoice PDF files are going to be stored in non-public folder. Special action serve is going to provide demanded PDF files addressed by special code, file code. This code is going to be included in new model FreeInvoiceStamp with attributes:

- vin_company – user's company VIN
- vin_division_company – users'
- price – total price of invoice
- file_name – name of PDF file, file_code + ".pdf"
- file_code – unique file code,
created as followed: `Digest::MD5.hexdigest(Time.now.to_s)`
- created_at – timestamp
- updated_at – timestamp

On Figure 24 is displayed screenshot of new open invoice form implementation. Companies data are already prefilled from ARES system. In mother company part the building number field is marked red as required. This demonstrates how JavaScript form validation works if fields are not filled (In this company's case ARES did not return building number, it is part of street field.). Prepared open invoice screenshot is displayed on Figure 25. To make the GUI ergonomic and transparent invoice forms and shows keep the same layout as output invoice file.

On Figure 26 is displayed example of output invoice file.

6.5.3 Statistics

This section should extend dashboard. Implementing statistics to the system should provide user a smart tool for evaluating his business. System is going to calculate economic indexes for chosen period of time. That is going to strengthen system's technical mastery as logic behind solving mathematical calculations is going to be harder to copy.

After discussing with users of the system (people close to GravaStar company) what indexes would they appreciate in statistics section and discussion with development team we decided to include following indexes:

- Turnover
- VAT balance
- Incomes total
- Expenses total
- Cashflow
- Profit/Loss

On the top of dashboard are going to be displayed four main indexes for current month. All indexes are going to be calculated for selected year, quarter, and month. Dashboard will also include system news and important states in the system.

6.5.3.1 Solution Design

Most important part of designing statistics is indexes definition. To provide useful information is crucial. In order to do so it was necessary to discuss indexes definitions with company's executive director and company's accountant. Final definitions of indexes are in Table 6.

Index	Definition
Turnover	= invoices sum price without VAT
VAT balance	= invoices vat in – expenses vat out = = (invoices turnover with VAT – invoices turnover) – (expenses sum VAT) = = ((invoices sum price with VAT) – (invoices sum price without VAT)) – (expenses sum VAT)
Incomes total	= sum of incomes price
Expenses total	= sum of expenses price
Cashflow	= incomes sum price – expenses sum price with VAT
Profit/Loss	= invoices turnover with VAT + not invoice incomes sum total price – expenses sum price with VAT

Table 6 Statistics indexes definitions

To display statistics is suggest separate year, quarter and month statistics in tabs. Each tab would consist of two parts displaying same indexes but for different period, e.g. year statistics tab will contain statistics from year 2012 and 2011. Each area will be navigated separately forward and backward in proper time period. Scale of the statistics is from time of a first entry until current date.

6.5.3.2 Solution Implementation

As some statistics were already implemented, not time scalable and with different indexes, I had to keep conventions of processing indexes. Statistics data are joined in one hash grouped by model of their origin. In every model is defined class method `dashboard_statistics` that provides data (returns hash), example of final calling statistics in dashboard's index action:

```
@stats = {:invoices =>
  invoice.dashboard_statistics(current_user),
  :expenses => Expense.dashboard_statistics(current_user),
  :incomes => Income.dashboard_statistics(current_user)}
```

`Dashboard_statistics` method calculates statistics for all time periods at once. This requirement causes longer loading time while serving AJAX requests for navigating in each statistic for time period. For navigation purposes `dashboard_statistics` has multiple input parameters, example of method header:

```
def dashboard_statistics(user, year = Date.today.year,
  quarter = 1, month = Date.today.month)
```

For demonstration how statistics are computed calculation of year statistics is examined. From `dashboard_statistics` is called another class method `year_statistics(user, year)`; current user and desired year are passed as attributes. I designed `year_statistics` as follow:

```
# returns hash with current year statistics
def year_statistics(user, year)
  invoices = self.all_for(user).from_year(year).summable
  turnover = 0
  vat_turnover = 0
  invoices.each do |invoice|
    turnover += invoice.price
    vat_turnover += invoice.price_with_vat
```

```

end
return {:turnover => turnover, :vat_in => vat_turnover -
turnover, :turnover_with_vat => vat_turnover}
end

```

To select proper set of invoices a set of named scopes is used. Scope is a model class method allowing executing specific queries on its model. ActiveRecord allows connecting scopes enabling wider usage of designed scopes. To select invoices in method `year_statistics` combination of three scopes is used. First `all_for(user)` selects only invoices that belongs to users mother company, `from_year(year)` selects only invoices issued in desired year, and `summable` eliminates canceled and credited invoices. I designed `from_year(year)` scope as follow:

```

named_scope :from_year, lambda {|year|
  {:conditions => ["printed_on >= ? and printed_on <= ?",
(Date.new year, 1, 1), (Date.new year, 12, 31)]}
}

```

Statistics are rendered all at once but only those in active tab are visible on dashboard page. Navigation process updates only selected part rest of statistics are still operational even can be navigated as well as navigation is done by AJAX call. Implemented statistics are showed on Figure 15.

Year statistics		quarter statistics		Month statistics	
Year 2011		Year 2010			
Turnover	492 692.00 Kč	Turnover	1 748 942.00 Kč		
VAT bilance	65 537.79 Kč	VAT bilance	236 729.03 Kč		
Incomes total	941 870.00 Kč	Incomes total	1 734 083.00 Kč		
Expences total	715 168.13 Kč	Expences total	1 391 342.48 Kč		
Cashflow	226 701.87 Kč	Cashflow	342 740.52 Kč		
Loss	-117 467.73 Kč	Profit	678 769.52 Kč		

Figure 15 Implementation of statistics

Static render of month statistics of selected indexes: Turnover, Incomes with VAT, Expenses with Vat, and Cashflow is placed in top part of dashboard page. Published actualities are displayed as a list and jQuery Carousel⁷ to slide automatically. The same plugin is used to slide list with information about unpaid invoices and collectable invoices. Redesigned dashboard with new statistics, actualities, and invoice information is in attached Figure 27.

6.5.4 Signature Image Upload

System provides possibility to print user's signature image on invoice output template. Its usage is simple user has to provide image with signature and upload it to the system. Then it is a matter of setting whether to sign issued invoices.

Image upload part lays a lot of responsibility on user, as image has to meet certain requirements to be suitable for placement into invoice. User has to prepare image at his computer and then upload to the system.

6.5.4.1 Solution Design

As analysis of competing products revealed most of systems do use the similar way of uploading signature image. They also require user to prepare signature image by certain rules.

The system now specifies image requirements: "Picture should be in PNG and 300 DPI" as is showed on Figure 16.

User's data	
Name	<input type="text" value="Jan Křička"/>
E-mail	<input type="text" value="xkrij117@czu.cz"/>
OpenID identity address	<input type="text"/>
Signature image	<input type="button" value="Choose File"/> no file selected

Picture should be in PNG and 300 DPI

Figure 16 User's data edit form

I propose including set of instructions how to easily achieve creating such image with brief description why is that. It is in the user's best interest to achieve upload of proper signature image.

⁷ jQuery Carousel [online], [25.11.2012], http://www.thomaslanciaux.pro/jquery/jquery_carousel.htm

I checked requirements⁸ for upload ID photo to Prague's Opencard where they specify accepted image content types, maximum uploaded size, width and height of image in pixels (with tolerance). Important parameters of the image are mentioned: dimensions. Purpose why certain DPI (Dots per inch) is required is to preserve signature's real dimensions, so it will have the same size on issued invoice as it user will make it himself. There is no limitation stated (e.g. in centimetres). Will the system print properly invoice with signature of size 20x20cm? The system cannot process large images because they break invoice's rendering template. To solve this issue I propose to set signature image to lower layer than text in invoice rendering template.

6.5.5 Invoice Filtering

Invoices are ordered by date and time of issuing descendant. That means that last issued invoice is going to be on the top of the list. Invoice list is paginated by 30 invoices per page and older invoices are hidden in further pages.

Add filtering for invoices was selected as feature that will improve user's interaction with system. It can shorten time spent on browsing the list.

6.5.5.1 Solution Design

Filtering lists is well known issue. In almost every application is implemented such feature. System already offers filtering invoices by state: all, unpaid, and delayed. Proposed solution is to add filters also to invoice attributes. Make possible to sort list according to one parameter in selected direction, order: asc (ascending), desc (descending). After discussion with project development team these invoice attributes were selected for filtering:

- Number
- Symbol
- Printed on
- Collectable on
- Price

6.5.5.2 Solution Implementation

To be able to sort invoices named scopes were created for each selected attribute and its direction. If parameters for sorting is send to invoice list action proper scope is applied to invoice collection. To make method more time efficient content of sorting parameter is directly called on invoice model

⁸ https://portal.opencardonline.cz/Download/povinne_nalezitosti_prukazove_fotografie.pdf

using ActiveRecord method send(). Content of the order parameter is designed to correspond with named scope names, example of calling scope:

```
invoices_unpaid =  
Invoice.all_for(current_user).billed_to(@company).billed_by(@  
division).unpaid.send(params[:sort])
```

In invoice list table header two small arrows were placed next to selected parameters. Top arrow orders list ascending, lower arrow orders list descending according to attached attribute. Implementation of invoice filtering is showed in Figure 17.

Logged user: Jan Křížka [LOGOUT](#)

nis DASHBOARD **INVOICES** EXPENSES INCOMES COMPANIES DIVISIONS TAX RETURN PROFILE

Listing invoices

Unpaid 87 922.00 Kč **Delayed 77 922.00 Kč**

[+ New invoice](#) State Year

no.	SYM	Printed	Collectable	Paid	Method	Price	Price with VAT	
0801038	CE	2008-10-09	2008-10-23	2008-10-29	Bank transfer	249 000.00 Kč	296 310.00 Kč	DETAIL PDF

Figure 17 Invoice filtering implementation

Filtering was extended to incomes list where is possibility to sort by date of payment and price. While sorting according date of payment unpaid incomes are always on the top of the list.

Companies list was also extended with filtering feature. This list can be ordered by name and company's invoice count.

7 Conclusion

Aim of this diploma thesis was to analyse and design solutions, or provide implementation of designed solutions, of online invoicing system operated by company GravaStar s.r.o. This thesis should concern given areas of research in order to increase number of registered users of the system.

I briefly described the system and analyse it. Analysis consist of business SWOT analysis that told us how the product stands on the market and measuring software product quality of the system describing system performance and qualities. Combining these two points of view pointed out things that the system's development team and me should focus on. This led to forming specification of things designed in given research areas.

For each area of research I provided introduction to issue and designed solution that will in my opinion improve system functionality and usability and are in compliance with formed specifications.

Some of research areas covered large scale of work. Among such areas I would place registration process, free invoicing for non-registered users as a marketing tool, and statistics. I implemented all designed solutions of these areas. Invoice filtering was also implemented. Designed solution was not properly discussed with system development team therefore it rested in design stage.

Registration process, statistics, and invoice filtering implementation are tested and released in production version, which means that users are already using my solutions. Open invoice is in testing phase. It is functioning properly, according to specification, but as big change in session management was made it still waits for confirming its functionality in production environment.

Future continuation of this work I see in measuring efficiency of implemented changes including releasing open invoicing to production. How the changes are going to reflect in amount of new users? Statistics about users can be gained by examining system database or new module mapping administration statistics can be designed. Users should be encouraged to give feedback about these changes. Users can answer questioners targeting specific areas or get interact with system's profile on social network. In this system lifecycle there is always going to be space for design new or redesign existing service or feature.

From external company's point of view I think this work is beneficial. New component extending system's provided services was designed and implemented. Other main research areas were implanted and released into production environment. Provided analysis can be used for further system development.

I gained new experiences with software development, for example management responsibility associated with preparing materials for system analysis and evaluation, and following solution design and implementation. Planning all phases of project of this scale I consider as beneficial for me.

8 References

8.1 Literature

COERS, Mardi, GARDNER, Chris, Benchmarking: A Guide for Your Journey to Best-Practice Processes, American productivity & Quality Center, 2004, ISBN 1-928593-24-0

COLLIS, David J., MONTGOMERY, Cynthia A., Competing resources, Harvard Business Review, July – August 1995

FULTON, Hal, The Ruby Way, SAMS, 2002, ISBN-10: 0-672-32083-5

GROVE, Ralph. F, Web Based Application Development, Jones and Barlett Publishers, 2009, ISBN-10: 0763759406

HARVARD BUSINESS SCHOOL PRESS, Strategy: Create and Implement the Best Strategy for Your Business, Harvard Business School Press, 2005, ISBN-10: 1591396328

KEDAR, Seema, Database Management System, Technical Publications Pune, 2009, ISBN 9788184316049

MILLER, Phillip M., TCP/IP - The Ultimate Protocol Guide: Volume 1 - Data Delivery and Routing, Brown Walker Press, 2009, ISBN-10: 1-59942-491-6

O'BRIEN, James, MARAKAS, George, Introduction to Information Systems, 15th edition, McGraw-Hill/Irwin, 2009, ISBN-10: 0073376779

OGC - OFFICE OF GOVERNMENT COMMERCE, Service Design, United Kingdom for The Stationary Office, 2007, ISBN 978 0 11 331047 0

OGC - OFFICE OF GOVERNMENT COMMERCE, The Official Introduction to the ITIL Service Lifecycle, United Kingdom for The Stationary Office, 2007, ISBN 9780113310616

OLSEN, Russ, Eloquent Ruby (Addison-Wesley Professional Ruby), Pearson Education Inc., 2011, ISBN-10 0-321-58410-4

PARSONS, David, Dynamic Web Application Development Using XML and Java, Cengage Learning EMEA, 2008, ISBN-10: 1844805417

RAINER, R. Kelly, TURBAN, Efraim, Introduction to Information Systems: Supporting and Transforming Business, 2nd Edition, 2008, ISBN 0470169001

RUBY, Sam, THOMAS, Dave, Agile Web Development with Rails (3rd edition), The Pragmatic Bookshelf, 2009, ISBN: 978-1-93435-616-6

THOMAS, Dave, FOWLER, Chad, Programming Ruby (2nd edition): The Pragmatic Programmers' Guide, The Pragmatic Bookshelf, 2009, ISBN: 978-0-9745-1405-5

VANICEK, Jiri, PAPIK Martin, Information systems quality rating, Czech university of life sciences in Prague, 2012

VRANA, Ivan, RICHTA, Karel, Zásady zavádění podnikových informačních systémů, 1st edition, Grada Publishing, 2005, ISBN 80-247-1103-6

YEAGER, Nancy J., MCGRATH, Robert E., Web Server Technology: Advanced Guide for World Wide Web Information Providers, Morgan Kaufman Publishers Inc., 1996, ISBN: 9781558603769

8.2 Electronic Documents

Browser Statistics [online], [quot. 8.11.2012], w3schools, http://www.w3schools.com/browsers/browsers_stats.asp

Definition of analysis [online], [quot. 19.11.2012], Oxford Dictionaries, <http://oxforddictionaries.com/definition/english/analysis?q=analysis>

FuzzyFinder [online], [quot. 9.11.2012], FuzzyFinder, http://www.vim.org/scripts/script.php?script_id=1984

Firebug [online], [quot. 9.11.2012], Firebug, <https://www.getfirebug.com/>

Git [online], [quot. 9.11.2012], Git, <http://git-scm.com/>

GitX [online], [quot. 9.11.2012], GitX, <http://gitx.frim.nl/>

Mozilla Firefox Web Browser [online], [quot. 8.11.2012], Mozilla.org, <http://www.mozilla.org/en-US/firefox/new/>

MySQL [online], [quot. 9.11.2012], MySQL, <http://www.mysql.com/why-mysql/>

Nuis [online], [quot. 10.11.2012], Nuis, <http://www.nuis.cz/>

RubyGems.org [online], [quot. 9.11.2012], RubyGems.org, <http://rubygems.org/>

Vim [online], [quot. 9.11.2012], Macvim, <http://macvim.org/OSX/index.php>

Web application framework [online], last modified on 8 November 2012 at 16:02 [quot. 8.11.2012], http://en.wikipedia.org/wiki/Web_application_framework

WEBrick [online], [quot. 9.11.2012], GitHub, https://github.com/nahi/ruby/tree/webrick_trunk

Web browser [online], last modified on 7 November 2012 at 17:35 [quot. 8.11.2012], Wikipedia, http://en.wikipedia.org/wiki/Web_browser

III. Attachements

9 Attached Images

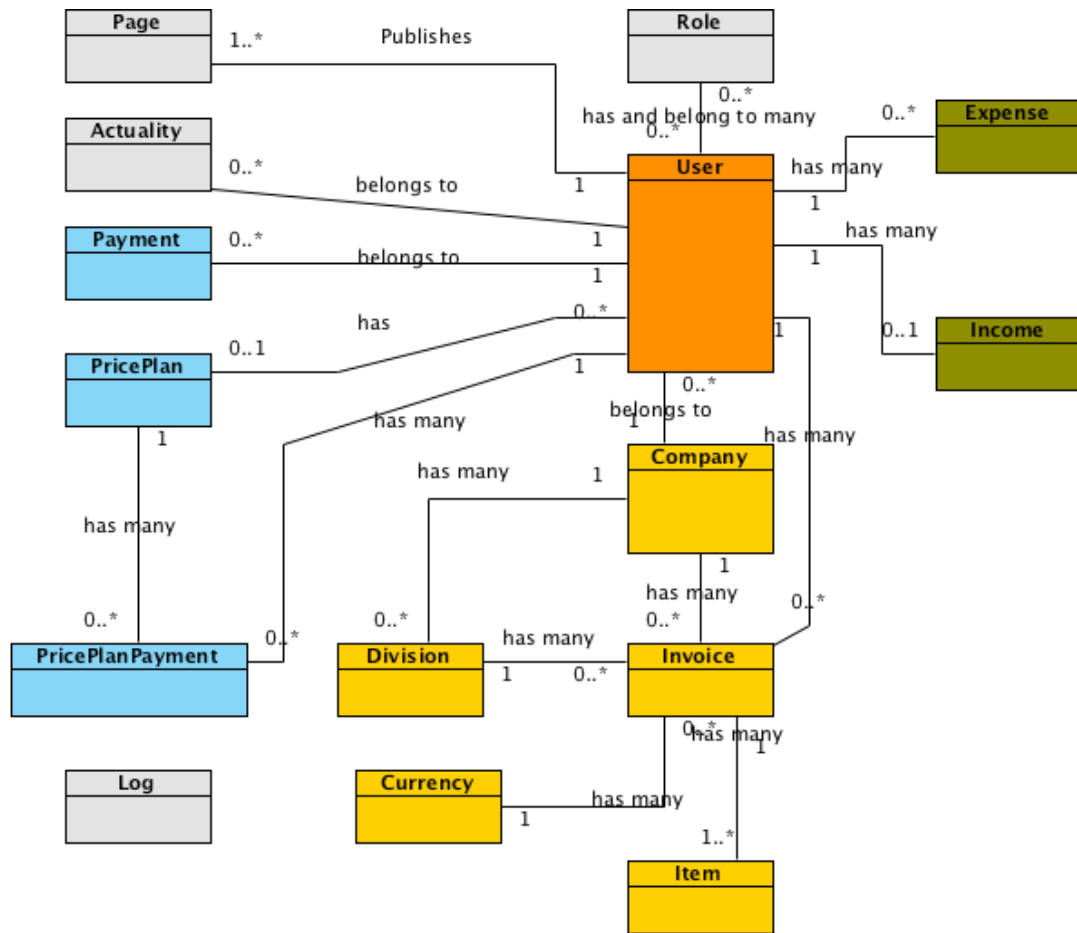


Figure 18 System models in UML diagram

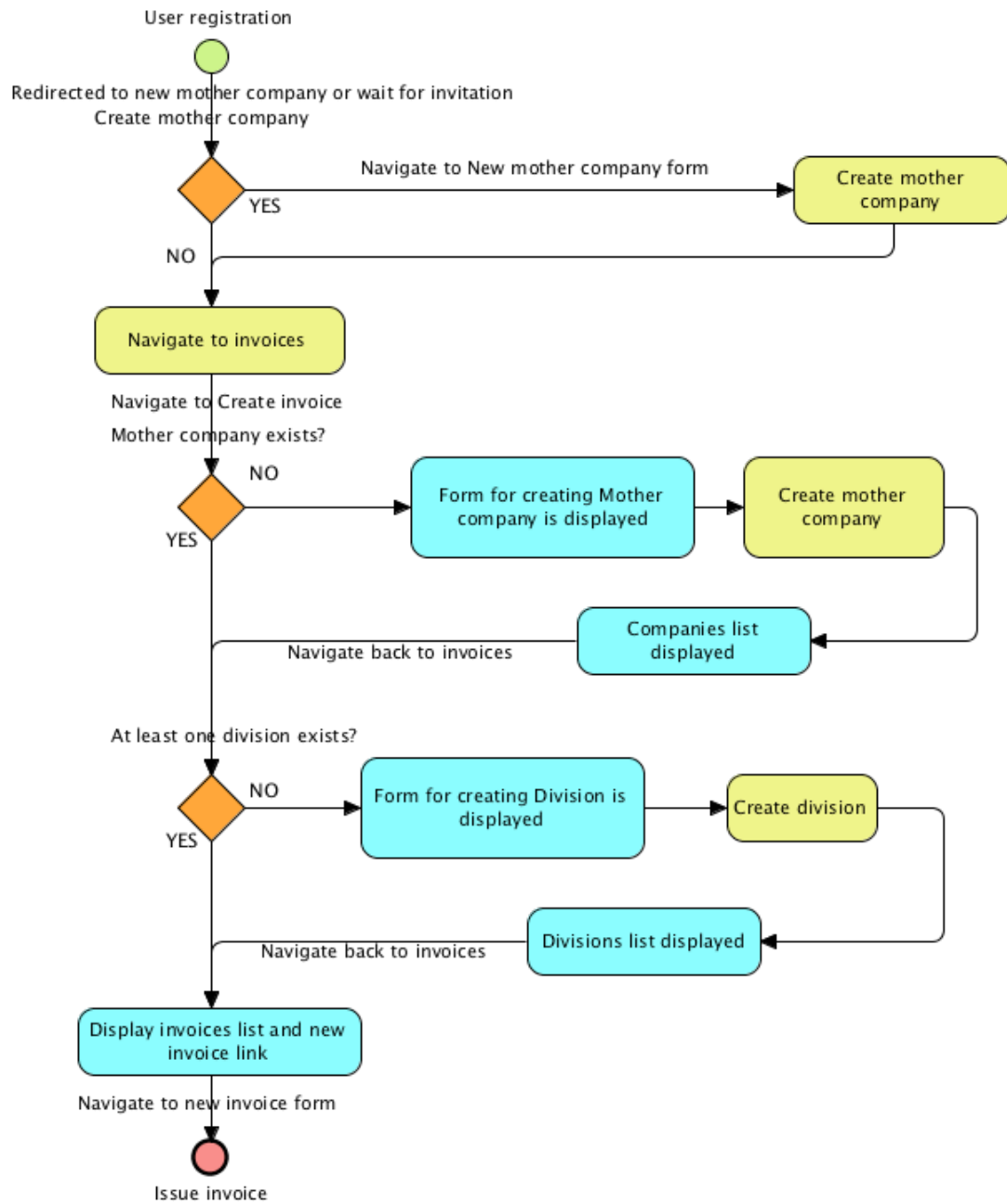


Figure 19 First invoice - initial state

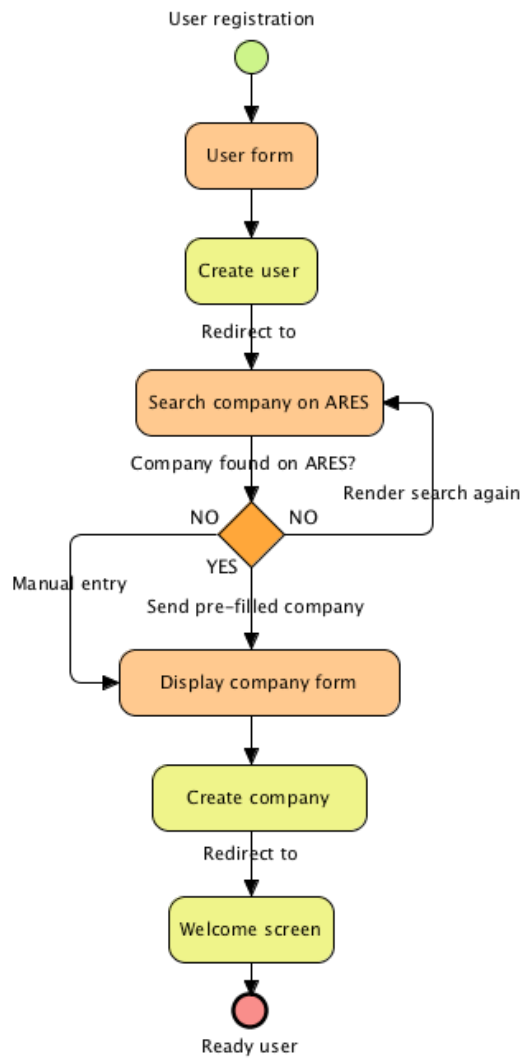


Figure 20 Proposed registration process workflow

The image shows a web registration form for 'njs'. At the top left is the 'njs' logo. Below it, a progress bar shows two steps: '1 Create new account' (active) and '2 Next information'. The main heading is '1 Create new account'. The form fields are: 'Programme' with a dropdown set to 'FREE' and a note 'Choose price plan.'; 'E-mail' with the value 'xkrij117@czu.cz', a red star icon, and a note 'Enter a valid email address ex. helpdesk@nuis.cz.'; 'Name' with the value 'Jan Kricka', a red star icon, and a note 'Enter your name ex. Pavel Novak'; 'Login' with the value 'xkrij117@czu.cz', a red star icon, and a note 'Enter name you will login to the system.'; 'Password' with a masked field of seven dots, a red star icon, and a note 'Enter password. At least 6 characters long.'; 'Password confirmation' with a masked field of seven dots and a red star icon; and 'Coupon' with an empty text box. At the bottom left, there is a checked checkbox and the text 'I agree with [business conditions](#)'. At the bottom right, there is a dark grey button labeled 'Next information'.

Figure 21 User part of registration process

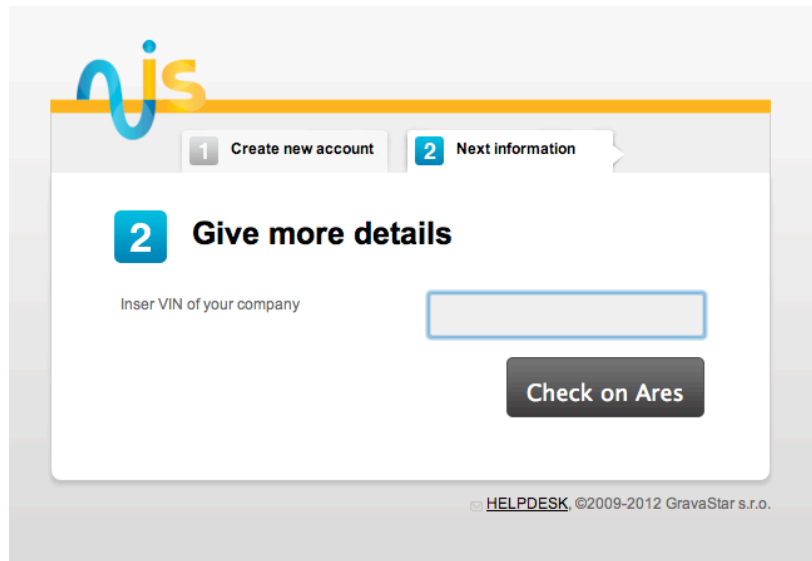


Figure 22 Company part of registration process - checking ARES



1 Create new account

2 Next information

2 Give more details

Company/Businessman detail

Name	<input type="text" value="ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA"/>	Enter the business name.
Symbol	<input type="text"/>	★ Symbol is used for easier finding companies. Fill in (own) abbreviation for company.
Street	<input type="text" value="Kamýcká 129"/>	★ Enter street name without house number ex.: Charles de Gaulle
Building number	<input type="text"/>	★ Enter house number ex.: 800/3
City	<input type="text" value="Praha-Suchdol"/>	★
POSTAL CODE	<input type="text" value="16000"/>	★ Enter a zip code in the form: 16000
Country	<input type="text"/>	★ ex: Czech Republic
Country code	<input type="text" value="cz"/>	★ Enter the country code ex.: cz
or sk		
VIN	<input type="text" value="60460709"/>	★ back to search by VIN on ares
VRC	<input type="text"/>	
Payer of vat	<input type="radio"/> yes <input checked="" type="radio"/> no	
Incorporate	<input type="text"/>	★
It will be a picture of the invoice. Ex.: Společnost GravaStar s.r.o. je zapsaná v obchodním rejstříku, vedeném Městským soudem v Praze, oddíl C, vložka 115629.		
E-mail	<input type="text" value="xkrij117@czu.cz"/>	★ Company email.
Account number	<input type="text"/>	Enter in format with a slash: 1234567890/0800.
Company note		

Save

Figure 23 Company part of registration process



New invoice

Mother company detail

VIN for ARES

Name

Vin

Vrc

Phone

E-mail

Account number

Street

Building number

Please enter this field

City

Postal code

Payer of vat yes no

Company detail

VIN for ARES

Name

Vin

Vrc

Street

Building number

City

Postal code

Country

Number and currency

Invoice number

Currency

Dates

Printed

Collectable

Taxable payment on

Company data

Ordered by

Method

Items

Note

Name	Amount	Price per unit	Price	VAT percent	Options
Scholarship	1	10000		20 %	
<input type="text"/>				20 %	
<input type="text"/>				20 %	

Invoice symbols

Order number

Variable symbol

Constant symbol

Specific symbol

Figure 24 New open invoice form



New invoice

Mother company detail

Name ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE
Vin 60460709
Vrc
Phone
E-mail
Account number
Street Kamýčká
Building number 129
City Praha-Suchbát
Postal code 16000
Payer of vat yes

Company detail

Name GravaStar s.r.o.
Vin 27386830
Vrc
Street Podbabská
Building number 81/17
City Praha 6
Postal code 16000
Country Česká republika

Number and currency

Invoice number 007
Currency Kč

Dates

Printed 2012-11-24
Collectable 2012-12-08
Taxable payment on 2012-11-24

Company data

Ordered by Jan Kricka
Method Bank transfer

Items

Name	Amount	Price per unit	Price	VAT percent
Scholarship	1	10 000.00 Kč	10 000.00 Kč	20 %

Invoice symbols

Order number
Variable symbol 117
Constant symbol
Specific symbol 1234567

Total info

Total 10 000.00 Kč
Total price incl. vat 12 000.00 Kč

Edit invoice

Issue invoice

[HELPDESK](#) ©2009-2012 GravaStar s.r.o.

Figure 25 Prepared open invoice

Invoice no.: 007

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V
PRAZE
Kamýcká 129
16000 Praha-Suchdol

Order number:
Variable symbol: 117
Constant symbol:
Specific symbol: 1234567

Vin: 60460709
Vrc:
Phone:
E-mail:

Account number:
Printed: 2012-11-24
Collectable: 2012-12-08
Taxable payment on: 2012-11-24

Method: Bank transfer

Customer:

GravaStar s.r.o.
Podbabská 81/17
16000 Praha 6
Česká republika

Vin: 27386830
Vrc:

Ordered by: Jan Kricka

Name	Amount	Price per unit	Price	VAT percent	Price with VAT
Scholarship	1	10 000.00 Kč	10 000.00 Kč	20%	12 000.00 Kč

total: 10 000.00 Kč
VAT 20%: 2 000.00 Kč
Total price incl. VAT: 12 000.00 Kč

Printed by:

Generated by NUIS.cz

Figure 26 Open invoice PDF output

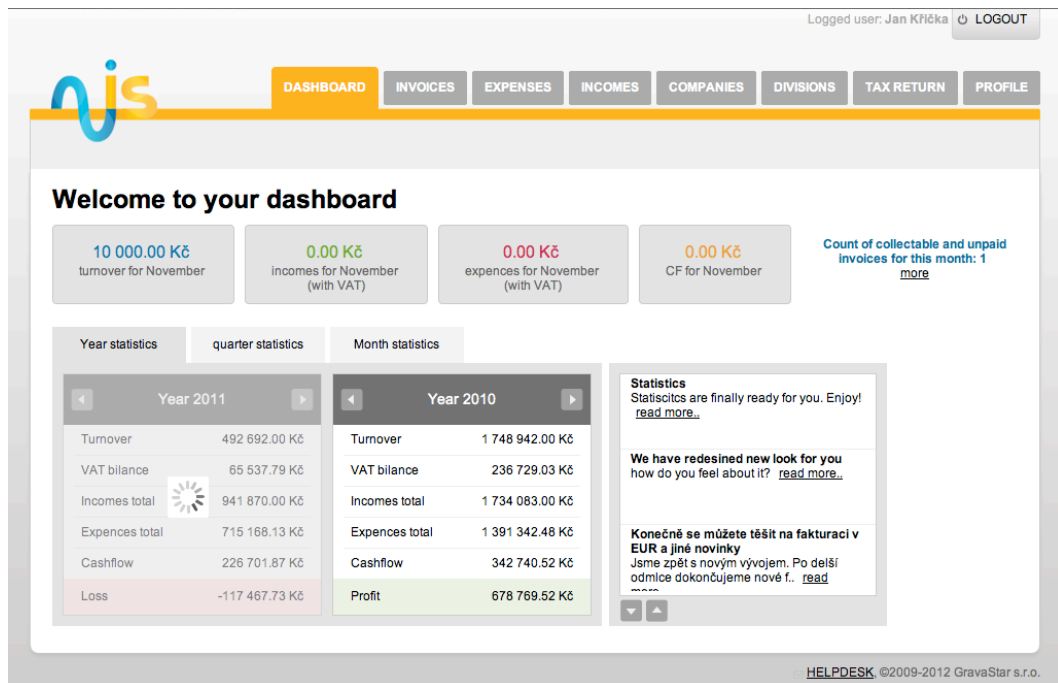


Figure 27 Redesigned dashboard

10 Attached Files

10.1 20110211105948_add_freeze_attributes_to_invoice.rb

```
class AddFreezeAttributesToInvoice < ActiveRecord::Migration
  def self.up
    add_column :invoices, :division_company_name, :string
    add_column :invoices, :division_company_street, :string
    add_column :invoices, :division_company_building_number, :string
    add_column :invoices, :division_company_city, :string
    add_column :invoices, :division_company_postal_code, :string
    add_column :invoices, :division_company_country, :string
    add_column :invoices, :division_company_account_number, :string
    add_column :invoices, :division_company_vin, :string
    add_column :invoices, :division_company_vrc, :string
    add_column :invoices, :division_company_phone, :string
    add_column :invoices, :division_company_email, :string
    add_column :invoices, :company_name , :string
    add_column :invoices, :company_street, :string
    add_column :invoices, :company_building_number, :string
    add_column :invoices, :company_city, :string
    add_column :invoices, :company_postal_code, :string
    add_column :invoices, :company_country, :string
    add_column :invoices, :company_vin, :string
    add_column :invoices, :company_vrc, :string
    add_column :invoices, :division_name, :string
    add_column :invoices, :division_incorporate, :string
    add_column :invoices, :division_logo_image, :string
    add_column :invoices, :printed_by_name, :string
    add_column :invoices, :printed_by_stored_filename_path, :string
    add_column :invoices, :not_payer_of_vat, :boolean
  end

  def self.down
    remove_column :invoices, :division_company_name
    remove_column :invoices, :division_company_street
    remove_column :invoices, :division_company_building_number
    remove_column :invoices, :division_company_city
    remove_column :invoices, :division_company_postal_code
    remove_column :invoices, :division_company_country
    remove_column :invoices, :division_company_account_number
  end
end
```

```
remove_column :invoices, :division_company_vin
remove_column :invoices, :division_company_vrc
remove_column :invoices, :division_company_phone
remove_column :invoices, :division_company_email
remove_column :invoices, :company_name
remove_column :invoices, :company_street
remove_column :invoices, :company_building_number
remove_column :invoices, :company_city
remove_column :invoices, :company_postal_code
remove_column :invoices, :company_country
remove_column :invoices, :company_vin
remove_column :invoices, :company_vrc
remove_column :invoices, :division_name
remove_column :invoices, :division_incorporate
remove_column :invoices, :division_logo_image
remove_column :invoices, :printed_by_name
remove_column :invoices, :printed_by_stored_filename_path
remove_column :invoices, :not_payer_of_vat
end
end
```