Vrije Universiteit Amsterdam,
Faculty of Sciences

USABILITY AND PRESENTATION LAYER OF A WEB BASED SALARY APPLICATION

Final Project Thesis 36 ECTS

AUTHOR: LAURA GARRIDO PURAS
SUPERVISOR: ANTON ELIËNS
CO-SUPERVISOR: PROFESOR DR. GERRIT VAN DER VEER

Amsterdam, Holland
2008

Acknowledgements

Dedicated to my wonderful parents, Domingo and Candi, sister, Irene, brother, Raúl and boyfriend, Jesús.

Personally, I would like to thank Dr. Anton Eliëns and Pf. Dr. Gerrit van der Veer for serving as my supervisor and co-supervisor, helping me through my thesis work. I truly appreciate all the time and advice you gave me. Thanks for treating me with respect and correcting and explaining all my errors during the development of this thesis in a way that I could understand and learn from them. I appreciate your guidance, support and willingness to take time to discuss my research.

Thank you to Luis Abreu for acting as a second advisor to me. Thanks for taking the time to teach me what you know and helping me throughout this thesis work. I wish I knew half as much as you! I have derived enormous personal and professional benefit from my time spent at Personeel en Zo, both from the people who work here and the environment that they have created. I really felt at home. Thank you to Adriaan, Arthur, Bayron, Bertjan, Colin, Danielle, David, Floris, Jeroen, Lonneke, Luis, Marcel, Marieke, Mervin, Raymond, Suzanne, Timon and Welmoed (notice the alphabetical order). I will never forget you.

Thanks to all my colleges at the faculty, but especially to Javi and Isma. We have done a very good team together all these years and I am sure that I would never have done most of the things I have done without your optimism and support. We have grown and learned together; we have cried, laughed and fought as well, but thank you for all that unforgivable moments.

I would like to thank my parents Domingo and Candi. Thanks for supporting me during my studies and helping me out on the bad moments. Thanks for your unflagging belief that despite your incomprehension about what I do, I am able to do it. Thanks for having blind faith on me and let me know that whatever I do I am going to do it well and you are going to be proud of me. Mom and Dad, I could not wish more wonderful parents; without you, it wouldn´t had been possible.
(Gracias por apoyarme en mis estudios y ayudarme en los malos momentos. Gracias por creer, a pesar de no terminar de entender lo que hago, que soy capaz de hacerlo. Gracias por tener fe ciega en mí y hacerme saber, que haga lo que haga y tome la decisión que tome, lo respetareis y estaréis orgullosos de mí. Mama, Papa, no podría desear unos padres más maravillosos. Sin vosotros no habría sido posible.)

Finally I want to thank my always loved boyfriend Jesús. Thanks for being always *on the other side.* Without you it would have never been possible. Thanks for sharing every minute with me and being always by my side. You have done everything easier and you have trusted on me without doubting. I didn´t know if I was going to make it, but now I know I can make everything if you are with me.
(Gracias por estar siempre *al otro lado.* Sin ti esto nunca habría sido posible. Gracias por compartir cada minuto conmigo y estar siempre a mi lado. Tú has hecho que todo fuera más fácil y siempre has confiado en mí sin dudar ni un momento. Has hecho que estando muy lejos, te sintiera muy cerca. No sabía si podría conseguirlo, pero ahora se, que si tu estas a mi lado, puedo conseguirlo todo.)

Preface

This thesis is the final work of my Informatics Engineering degree studied at the Polytechnic University of Madrid. The work described on it was carried out between February and August 2008 at Personeel en Zo BV, Amsterdam, during an ERASMUS scholarship.
The thesis consists of seven chapters and three appendixes. The first part is the introduction and background information, being the second part the main one in which the work is developed. There is also a final conclusion summarizing the whole process.

Table of Contents

List of Figures

Chapter 1. *Introduction*

1.1 Introduction

In this thesis, *Usability and Presentation layer of a web based salary application* the quality of user's experience when interacting with the Personnel and Salary application, PersoneelOnline, will be measured and improved.

Personeel en Zo BV, is a company which provides salary administration support and payslip calculations for third parties.
They provide a web based product, *PZ Web*, through which their customers are able to provide them with the necessary input data for the salary administration. Once Personeel en Zo BV has the information needed, takes care of all the salary calculations using an offline software tool of another company called *Cobra*. Finally, the output is (online) made available for the client.

The goal of the project this company is working for, is to build their own web based salary administration tool, PersoneelOnline, in which both the payroll specialists an users can access and in which all the salary calculations are automatically made without using any other tool but this one.

The goal of this thesis is to evaluate the usability of the current tools and build the new salary administration tool taking into account those evaluations and the evaluations made to the new system while it is being built.

The activities in this Thesis include tasks analysis, take interviews to the different stakeholders and make iterative prototypes that lead to the best possible interface.

1.2 Thesis structure

This thesis is structured in the following way:

Chapter 2. Application Domain. **Web based pay roll system**: This chapter is to place the user into the salary and web applications context. The section 2.2 explains what a salary and personnel management system is. Later on, in the section 2.3 is explained what is a web based application. Once the user knows about salary and web based applications, the section 2.4 explains the situation of the company who is working on this project, what led them to start it and why using a web based application. To finish, the section 2.5 talks about the new system being built explaining its main characteristics and features so the reader knows what is it about when they were mentioned along the following chapters.

Chapter 3. Background- Usability: This chapter is to set a usability background. It talks about what is usability and what does usability measure. The reader will be introduced on the scenario based usability engineering. Requirement analysis, prototyping and different forms of evaluation will be shown.

Chapter 4. Task analysis: In this chapter the analysis of current and future situation is made. Here is where the stakeholders are defined and where the interviews to the user take place. The Appendix A.1 shows the different roles a user can play on this application as well as the different user levels and the tasks a user can carry out depending on his user level and role. Although the interviews themselves are on the Appendix A.2 and the outcomes of those interviews on the Appendix A.3, a table on the chapter resumes them all.

Chapter 5. Prototyping: In this chapter are shown the main characteristics and evaluation of each prototype. The prototyping face is made in an iterative way as each prototype is first developed to later on make an evaluation over it to start with the following prototype. The evaluations are made on this order: first, the main characteristics of the prototype are mentioned. Later a list with the problems found during the evaluation is made and to finish, the solutions to those problems are exposed. In the Appendix B the main characteristics of

each prototype are addressed as well as several screenshots showing the problems and solutions of each prototype; and sometimes steps in between that lead to the final result.

Chapter 6. Technical Design: This chapter was born as the need of explaining some technical components that has been developed for this particular application, and that maybe can be used in others. Over the chapter the problem, the solution, which is the component itself, when it can be used and its implementation are described. On the Appendix C the code is shown. This code is only the client side one which can be reused no matter the framework used on the server side. It is not useful to show the server side code as it depends on what framework each developer uses.

Chapter 7. Conclusions: Finally, this chapter contains the concluding remarks.

Chapter 2. *Application Domain – Web based pay roll system*

2.1 Overview

This thesis is about a salary web based application called PersoneelOnline. This chapter explains what is a salary and personnel management system, what is a web based application, the situation of the company who is working on this project, what led them to start it and why using a web based application. The chapter finishes talking about the new system, PersoneelOnline, explaining its main characteristics and features.

2.2 Salary and Personnel Management System

Salary and Personnel Management Systems include the following modules:

1. Payroll
2. Work Time
3. Benefits Administration
4. HR management Information system

The **Payroll module** keeps track of all the employees´ information as sicknesses, vacation hours still available or overtime hours worked. It automates the pay process by gathering data on employee time and attendance, calculating the taxes and deductions, and generating periodic pay slips and taxes reports. Data is generally fed from the human resources. This module can encompass all employee-related transactions as well as integrate existing financial management systems.
The process to carry out the payment of the payroll is complex because we have to take in account different factors as the salary base, the social security, loans, etc. Each one of these aspects is contemplated in the system, which will help significantly to make the process easier.
Other advantages of this system is that permits the integration with technological tools as Internet, helps to resolve particularities of the client; the modules of finances and costs are integrated real-time offering more control, flexibility and reduction of costs by optimization of resources.
It provides a flexible solution that allows making automatic the process of payroll in any company of any size, controlling the information of the process of payroll with confidence, maximizing the profit value of the business and taking legal changes into account.

The **Work Time** gathers standardized time and work related efforts. The most advanced modules provide broad flexibility in data collection methods, labor distribution capabilities and data analysis features. Cost analysis and efficiency metrics are the primary functions.

The **Benefits Administration module** provides a system for organizations to administer and track employee participation in benefits programs. These typically encompass insurance, compensation, profit sharing and retirement.

The **HR management module** is a component covering many other HR aspects from application to retirement. The system records basic demographic and address data, selection, training and development, capabilities and skills management, compensation planning records and other related activities. Leading edge systems provide the ability to "read" applications and enter relevant data to applicable database fields, notify employers and provide position management and position control. Human resource management function involves the recruitment, placement, evaluation, compensation and development of the employees of an organization.
Initially, businesses used computer based information system to:

- Produce pay checks and payroll report.
- Maintain personnel records.
- Pursue Talent Management.

2.3 Web-Based Application

In software engineering[1] a Web application is an application that is accessed via Web over a network such as the Internet or an intranet.
It is used when the site is for "doing" things rather than finding information. Web applications are popular due to the ubiquity of a client. The ability to update and maintain Web applications without distributing and installing software on potentially thousands of client computers is a key reason for their popularity. A web-based application is an application that could just as well be a normal application. It now just runs in a web browser.

In earlier types of client-server computing, each application had its own client program which served as its user interface and had to be separately installed on each user's personal computer. An upgrade to the server part of the application would typically require an upgrade to the clients installed on each user workstation, adding to the support cost and decreasing productivity.

In contrast, Web applications dynamically generate a series of Web documents in a standard format supported by common browsers such as HTML/XHTML. Client-side scripting in a standard language such as JavaScript is commonly included to add dynamic elements to the user interface. Generally, each individual Web page is delivered to the client as a static document, but the sequence of pages can provide an interactive experience, as user input is returned through Web form elements embedded in the page markup. During the session, the Web browser interprets and displays the pages, and acts as the universal client for any Web application.

AJAX, which stands for "Asynchronous JavaScript and XML," helps make Web-based applications nearly as responsive as software that resides on a user's computer. The technique helps speed up computer operations by cutting down on the need to request fresh Web pages from a distant server computer. Instead, Ajax applications can request smaller chunks of data to update a Web page already on a user's screen.

In this particular case, the ASP.NET framework is used as end point on the client side. The following diagram shows how a web-application works. On it AJAX requests are used to make the communication between JavaScript and the server easier for developers.

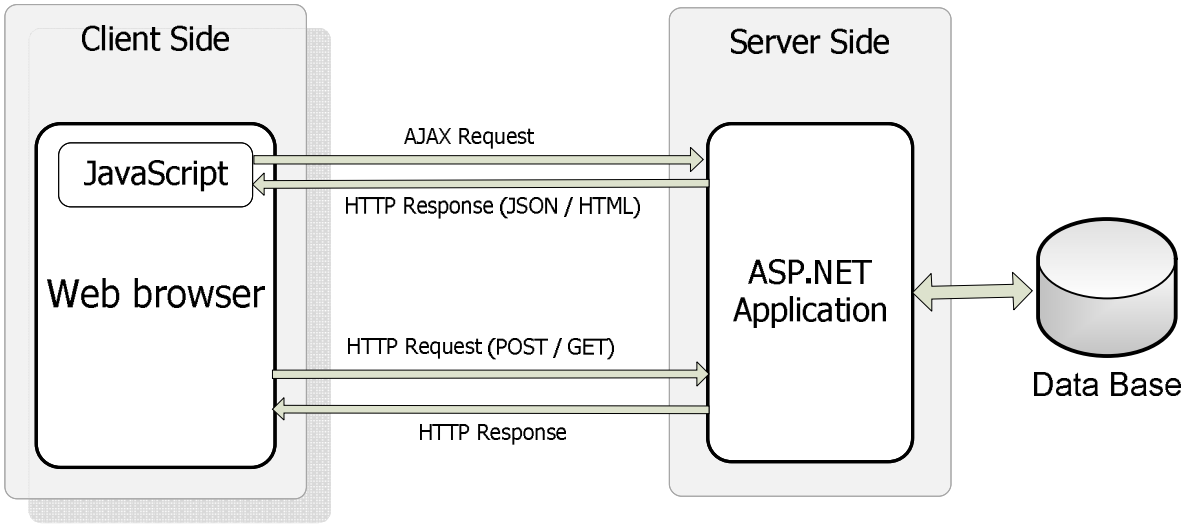


Figure 1: Web based application

[1] Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software.

2.4 Case Description Personeel en Zo BV

Personeel en Zo BV is a business specialised in the field of salary administration. They provide salary administration support and payslip calculations for third parties with the main focus on medium and small size enterprises. In the last couple of years Personeel en Zo BV has undergone a growth of on average 40% in turnover and expects to continue this growth the coming year as well.

The service of Personeel en Zo BV consists of providing a web based product called PersoneelOnline!, through which their customers are able to provide (online, at every moment desired) Personeel en Zo BV with the necessary input data for the salary administration like (new) employees, hours worked, salary, date of start and end dates of employment et cetera, after which Personeel en Zo BV takes care of the wage and tax calculations, creates company overviews like payment lists and provides a legal check-up of the administration. The output is (online) made available for the client and after client approval, the necessary data is sent to the tax authority and the client can book the data in his company administration. The total provided services vary per client from just making the calculations, providing administrative advice/support to taking care of the wage payments and tax reports.

For the actual calculations for the salary administration Personeel en Zo BV currently uses an (offline) software tool of another company, several fulltime employed salary administrators and an online application which through the use of a server enables the clients to write the administration data into an Access database. The salary administrator uses the offline software tool to take the necessary data from the database to make the wage, tax calculations and company overviews while providing the control on the administration and if necessary warn about for example input mistakes made by the client. The (new) data is written back into the database and the output is made available for the client.

In the current system the clients and the salary administrators are not able to simultaneously dispose over all the administration data because after a client has provided the (monthly) input data it will have to be generated by one of the salary administrators of Personeel en Zo BV using the (offline) software tool before it can be made available online for the client to approve. This means that after the calculation and control by the salary administrator there might still be some errors in the output as a result of mistakes made during input by the client. One can think of faulty payslips because of a wrong number of days , hours worked by the employee or a false start date of a new employee filled in through PersoneelOnline!, which the salary administrator cannot identify. This results in double work and time lost for both parties, because some calculations have to be redone and send to the client for approval more than once before the payments and tax reports can finally be done.

Because in the current system intervention of the salary administrator is needed in order to generate the payslips, tax reports and company overviews the control on the salary administration is done during the calculation phase. But it is also possible to check and if necessary make corrections after the tax reports have been send to the tax authority. As a result of the control over the salary administration being done during the calculation phase, and as most clients do their salary administration and payments around the 20th of the month, there is a huge spike in the amount of workload for Personeel en Zo BV every month around the 20th. If calculation, overview and tax report generation could be separated from control, Personeel en Zo BV could spread the workload more evenly over the month and create an efficiency gain in their work process.

Also in the current system Personeel en Zo BV is heavily dependent on the (offline) software tool which they do not own and for which they have to pay to use. This makes Personeel en Zo BV as a company dependent on another company.
The strategic importance of owning a software tool of their own, together with the possible efficiency gain lead Personeel en Zo BV to the decision to develop their own (completely) web based salary administration tool. And I'll be part of the team, taking care of the usability and presentation layer for this new salary administration tool.

2.5 The new PersoneelOnline

As it is already known at this point, PersoneelOnline is a web based salary and personnel application.
This section talks about its main features so the reader knew what it is about when some of these features were mentioned on the following chapters.

Everything on this application is working under a *period*. The main goal of this application is generating pay slips and a pay slip belongs to a certain period of time. Three different types of periods can be found: a month period, a week period or a four weeks period. Sometimes along the thesis these period types will be abbreviated with *M* for a month period, *W* for a week period or *4W* for a four weeks period (as we can see on the application's header).

Each company will have to choose on which period type it wants to work. The monthly period type has 12 periods, the weekly period type can have 52 or 53 periods and the four weeks one 13 or 14. This way, a period must be always made up of a *year*, a *period* and a *period type*.

The period changes every time a payslip is generated.

This application gives the user the chance of changing the period being able to make changes into the future or the past. In the last case, if the user makes any change it could affect calculations or pay slips that have been already processed. And is because of that why this application has the need of including a *system mode* that can take the values of *Protected* or *Unprotected*. If the system is on protected mode, those changes won't be possible and the calculations won't be modified. But if the user is on unprotected mode, he will be allowed to make changes that can have side effects on these past calculations, and it will be always under the users' own risk.

The navigation of this application mainly appears on the left menu. This left menu consist of some panels in which the organizations, companies or/and employees are listed. By clicking on one of them (one organization, company or employee) all the information about it will be shown on the main content area.

Chapter 3. *Background - Usability*

3.1 What is Usability?

Usability is not just the appearance of the user interface (UI). Usability measures the quality of a user's experience when interacting with a product or system. In general, usability refers to how well users can learn and use a product to achieve their goals and how satisfied they are with that process. Usability means that people who use the product can do so quickly and easily to accomplish their tasks [Dumas and Redish, 1999]. Usability may also consider such factors as cost, effectiveness and usefulness.

3.2 What does usability measure?

It is important to realize that usability is not a single, one-dimensional property of a user interface. Usability is a combination of factors including [Dumas and Redish, 1999]:

- **Ease of learning** How fast can a user who has never seen the user interface before learn it sufficiently well to accomplish basic tasks? How easy it is to learn the main system functionality and gain proficiency to complete the job? This is usually assessed by measuring the time a user spends working with the system before that user can complete certain tasks in the time it would take an expert to complete the same tasks. This attribute is very important for novice users.

- **Efficiency of use** Once an experienced user has learned to use the system, how fast can he or she accomplish tasks? In other words, the number of tasks per unit of time that the user can perform using the system. The maximum speed of user task performance is looked for. The higher system usability is, the faster the user can perform the task and complete the job.

- **Memorability** If a user has used the system before, can he or she remember enough to use it effectively the next time or does the user have to start over again learning everything? It is critical for intermittent users to be able to use the system without having to climb the learning curve again. This attribute reflects how well the user remembers how the system works after a period of nonuse.

- **Error frequency and severity** How often do users make errors while using the system, how serious are these errors, and how do users recover from these errors? This attribute contributes negatively to usability. It does not refer to system errors. On the contrary, it addresses the number of errors the user makes while performing a task. Good usability implies a low error rate. Errors reduce efficiency and user satisfaction, and they can be seen as a failure to communicate to the user the right way of doing things.

- **Subjective satisfaction** How much does the user like using the system? This shows a users subjective impression of the system.

3.3 Scenario-Based Usability Engineering

Computers do more than just provide information and services for people to use. The design of computing systems is part of an ongoing cycle in which new technologies raise new opportunities for human activity; as people's tasks change in response to these opportunities, new needs for technology arise. The basic argument behind scenario-based methods is that descriptions of people using technology are essential in discussing and analyzing how the technology is (or could be) reshaping their activities. A secondary advantage is that scenario descriptions can be created before a system is built and its impacts felt.

A user interaction scenario is a story about people and their activities [Caroll & Rosson 1990]. Scenarios describe the setting, actor(s), and events of a user-computer interaction, but also include information about user's

mental activities (goals, plans, and reactions).Scenarios integrate the many tasks of system development by first organizing the analysis of user needs, and then serving as central representations of user needs that are developed in a systematic manner through design, evaluation, and documentation activities. It does not explicitly describe the use of software or other technological support to achieve a task. Using the vocabulary and phrasing of users means that the scenarios can be understood by the stakeholders[2], and they are able to participate fully in the development process. In fact, the construction of scenarios by stakeholders is often the first step in establishing requirements. Scenarios can be used to guide usability engineering - the scenario-based development (SBD) framework. The framework should not be understood as a waterfall, even though it flows from problem analysis to design and then to evaluation. At each step of the process, scenarios are analyzed and transformed in support of different development goals. It is assumed that all activities in SBD happen in an iterative and interleaved fashion, but for explanatory purposes they are organized into an idealized progression.

3.4 Requirements Analysis

In requirements analysis, the problem situation is studied through interviews with clients and other users (the stakeholders), field studies of the current situation, and brainstorming among users and developers. This input is used to formulate problem scenarios that convey important characteristics of the users, the typical and critical tasks they engage in, the tool they use, and their organizational context.

A key contribution of scenarios during requirements analysis is that they evoke reflection and discussion. Writing down a narrative of one situation almost immediately raises questions about other situations, about why this situation works the way it does, and how others situations might work differently. The concrete and narrative character of scenarios also facilitates mutual understanding and communication among the different groups who participate in requirements analysis.

There are two aims. One aim is to understand as much as possible about the users, their work, and the context of that work, so that the system under development can support them in achieving their goals. Building on this, our second aim is to produce, from the needs identified, a set of stable requirements that form a sound basis to move forward into thinking about design. This will be the requirements analysis' result: the **requirements specification,** a document that lists all functions and features that the proposed system must satisfy.

Conceptually, requirements analysis includes three types of activity:

- **Eliciting requirements**: communicating with customers and users to determine what their requirements are. This is sometimes also called requirements gathering. Several techniques can be employed to elicit the requirements from the customer. Historically, this has included such things as holding interviews, holding surveys, making recordings or just workplace observations.

- **Analyzing requirements**: the task of determining whether the stated requirements are incomplete, unclear, ambiguous, or contradictory, and then resolving these issues.

- **Recording requirements**: Requirements may be documented in various forms, such as natural language documents, HTA[3], use cases, user stories, or process specifications.

3.4.1 Different kinds of requirements

In software engineering, two different kinds of requirements have traditionally been identified: functional requirements, which say what the system should do, and non-functional requirements, which say what constraints there are on the system and its development.
A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that show how a use case is to be fulfilled. It is a requirement that, when satisfied, will allow the user to perform some kind of function. They are supported by

[2] The stakeholders name the groups or individuals who should be consulted or observed in the fieldwork
[3] Hierarchical Task Analysis

non-functional requirements, which impose constraints on the design or implementation (such as performance requirements, security, or reliability).

However, instead of referring to all requirements that are not functional as simply "non-functional" requirements, this could be refined into further categories:

- **Data requirements** capture the type, volatility, size/amount, persistence, accuracy, and value of the amounts of the required data.

- **Environmental requirements** refer to the circumstances in which the interactive product will be expected to operate. For example, what technologies will the product run or need to be compatible with and what technological limitations might be relevant.

- **User requirements** capture the characteristics of the intended user group.

- **Usability requirements** capture the usability goals and associated measures for a particular product.

3.4.2 Main techniques and steps in analyzing requirements

Stakeholders' identification

Stakeholder analyses are now arguably more important than ever because of the increasingly interconnected nature of the world. Choose any public problem and it is clear that 'the problem' encompasses or affects numerous people, groups and organizations. Figuring out what the problem is and what solutions might work are actually part of the problem, and taking stakeholders into account is a crucial aspect of problem solving. At a minimum, stakeholder analyses should help us figure out who the key stakeholders are and what would satisfy them. The technique used in this particular step could be brainstorming the list of potential stakeholders.

Preparing for the Field Study

Questions about the current situation the system is hoping to address will come up as the root concept is been developed. Now the different stakeholders have been identified, different guides for conducting our interviews with each stakeholder can be developed. Each guide should support the questioning process. At the top there is a reminder about what the interviewer is trying to accomplish. The goal is to learn what the participants think about their own activities, so the guide should avoid specific and pointed questions early in the interview. Instead, begin with open-ended prompts that explore general background and how the interviewees think about their work. More specific questions are listed at the end, reminding the interviewer to address these issues if they have not yet been raised in the earlier discussion.
In addition to preparing an interviewing guide, it must be decided how to document the field observations. If the work setting involves considerable physical manipulation of objects, a videotape may be helpful. Otherwise, a small tape recorder can be used to record conversations or maybe some video screen capture software can be used.

Summarizing the field data

Here it is discussed several summary representations that can be used to organize the findings about a project's stakeholders and their activities.

Stakeholders
The observations and interviews for each stakeholder group are organized into stakeholder profiles. These profiles summarize the general characteristics of each group, and are based on the observations and interviews from the field study. The summaries are organized into background and expectations for the proposed system.

Task Analysis
Another set of summaries is developed to document the tasks of each stakeholder group. A simple list of tasks that were observed could be one way of summarizing the task analysis.
For tasks that have many steps, or that are particularly important in an activity, a hierarchical task analysis may be developed. This analysis decomposes complex tasks into subtasks. The following figure presents a

hierarchical task analysis. Each box in the task analysis diagrams represents a task step. Vertical lines indicate decomposition of a step into two or more subtasks; the subtasks are gathered together under the horizontal lines. Numbering indicates how a task is decomposed, and the plans show the logical ordering or dependencies among subtasks.

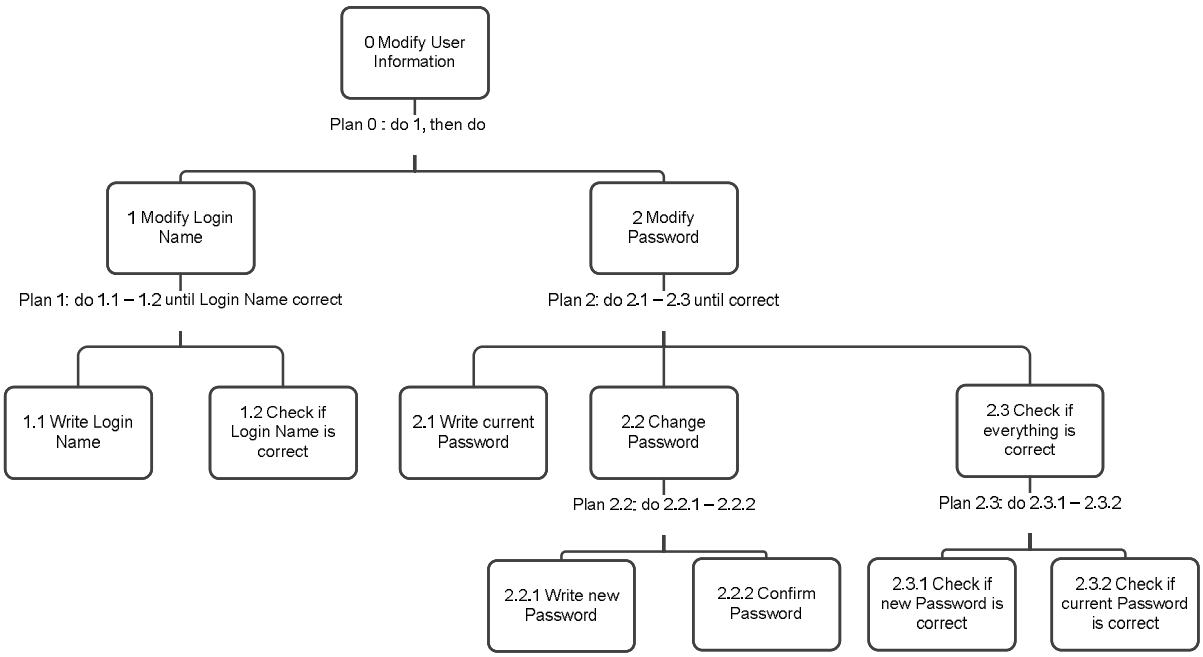


Figure 2: Hierarchical task analysis of Modify User Information

3.5 Prototyping

Prototypes are constructed and evaluated to guide redesign and refinement. A prototype is a concrete but partial implementation of a system design. A user interface prototype is a prototype built to explore usability issues [Wasserman & Shewmake 1982]

Usability testing is the core of usability engineering practice: Representative users are asked to interact with system prototypes and their behavior and subjective reactions are studied. Prototypes can be used to test all aspects of usability for a system – what users will expect when they encounter parts of the system, how they will go about pursuing their goals, how they will respond to system feedback, and what subjective reactions they will have.

The most convenient prototype for usability testing is an early working version of the system. A working system brings a sense of realism to test tasks. Task instructions can be minimal, because evaluators can rely on the system to guide users through the test tasks. Issues associated with overall complexity or internal consistency can be examined, and the measures of satisfaction or irritation that are collected will be much more meaningful.

Unfortunately, waiting for a running version of a system may mean that usability testing must be postponed well into the development process. A popular alternative is to build a realistic simulation with rapid prototyping tools (e.g., Visual Basic or Macromedia Director), with the understanding that the prototype is temporary and will be replaced eventually by the real system. A discardable prototype can be an excellent option if the usability professionals already have expertise with an appropriate tool, but if not, building the prototype may become a major implementation effort itself. There is also the risk that a prototype will exhibit enough functionality that the team comes to believe that what began as a prototype is the final system.

Some Advantages of Prototyping are [Vicki L. Sauter, 1999]:
- Reduces development cost.
- Requires user involvement.
- Developers receive quantifiable user feedback.
- Facilitates system implementation since users know what to expect.
- Results in higher user satisfaction.
- Exposes developers to potential future system enhancements.

Some Disadvantages of Prototyping are [Vicki L. Sauter, 1999]:
- Can lead to insufficient analysis.
- Users expect the performance of the ultimate system to be the same as the prototype.
- Can cause systems to be left unfinished and/or implemented before they are ready.
- Sometimes leads to incomplete documentation.
- If sophisticated software prototypes (CASE Tools[4]) are employed, the time saving benefit of prototyping can be lost.

Mockups and Proof-of-Concept Prototypes

Two principal prototype groups can be distinguished: mockups and proof of concept.

The purpose of a *mock-up* prototype is to understand the access and analysis requirements and the business activities behind them.
Mockups are also called *horizontal prototypes* because they take a single, horizontal picture of the application. They do not go deeply (or vertically) into the other layers of the application such as the business objects and the database layers. Mockups are a great way to determine whether the requirements are complete and understood. They also help validate the use cases, the navigational structure, and some of the logical interactions of the application.

A *proof-of-concept prototype* is meant to validate the requirements and confirm the technology recommendations and high-level design.
A proof-of-concept prototype is also called a *vertical prototype* because it looks at the application across the entire stack (UI, services, business objects, and database). They provide a reference for the development team on just how the system should work from top to bottom.
A proof-of-concept prototype is created by choosing a key requirement of the application and then building it out through each layer of the design.

3.6 Evaluation

Usability evaluation is a central activity in the usability process. It can determine the current version's usability level and whether the design works.

There are generally three types of usability evaluation methods: *Testing*, *Inspection*, and *Inquiry* [James Hom, 1998], [J. Nielsen, 1993.] [Dumas and Redish, 1999], [John Wiley & Sons], [H. Rex Hartson], [Lindgaard, 1994], [Nielsen, Jakob], [Rubin, Jeffrey].

Usability Testing

In Usability Testing approach, representative users work on typical tasks using the prototype and the evaluators use the results to see how the user interface supports the users to do their tasks. Testing methods include the following:

[4] Computer-aided software engineering (CASE) is the use of software tools to assist in the development and maintenance of software

Coaching Method: This technique can be used when the participants are allowed to ask any system-related questions of an expert tester who will answer to the best of his ability.
The purpose of this technique is to discover the information needs of users in order to provide better *training* and *documentation*, as well as possibly redesign the interface to avoid the need for the questions.

Co-discovery Learning: During a usability test, two test users attempt to perform tasks together while being observed. They are to help each other in the same manner as they would if they were working together to accomplish a common goal using the product. They are encouraged to explain what they are thinking about while working on the tasks. Compared to thinking-aloud protocol, this technique makes it more natural for the test users to verbalize their thoughts during the test.

Performance Measurement: This technique is used to obtain quantitative data about test participants' performance when they perform the tasks during usability test. This will generally prohibit an interaction between the participant and the tester during the test that will affect the quantitative performance data. It should be conducted in a formal usability laboratory so that the data can be collected accurately and possible unexpected interference is minimized. Quantitative data is most useful in doing comparative testing, or testing against predefined benchmarks. To obtain dependable results, at least 5 user participants are needed, while 8 or more participants would be more desirable.

Question-asking Protocol: During a usability test, besides letting the test users to verbalize their thoughts as in the thinking aloud protocol, the testers prompt them by asking direct questions about the product, in order to understand their mental model of the system and the tasks, and where they have trouble in understanding and using the system.

Remote Testing: Remote usability testing is used when testers are separated in space and/or time from the participants. This means that the testers cannot observe the testing process directly and that the participants are usually not in a formal usability laboratory. There are different types of remote testing. One is same-time but different-place, where the tester can observe the test user's screen through computer network, and may be able to hear what the test user says during the test through speaker telephone. Another is different-time different-place testing such as journal sessions, where the user's test session is guided and logged through a special piece of software as well as additional code added to the system being tested.

Retrospective Testing: If a videotape has been made of a usability test session, the tester(s) can collect more information by reviewing the videotape together with the user participants and asking them questions regarding their behavior during the test. So this technique should be used along with other techniques, especially those where the interaction between the testers and the participants is restricted. But using this technique means that each test takes at least twice as long. Another obvious requirement for using this technique is that the user's interaction with the computer needs to be recorded and replayed.

Shadowing Method: During a usability test, the tester has an expert user who explains him the behavior of the tested user. This technique is used when it's not appropriate for the test user to think aloud or talk to the tester while working on the tasks.

Teaching Method: During a usability test, let the test users interact with the system first, so that they get familiar with it and acquire some expertise in accomplishing tasks using the system. Then introduce a naive user to each test user. The Novice users are briefed by the tester to limit their active participation and not to become an active problem-solver. Each test user is asked to explain to the novice how the system works and demonstrate to him a set of pre-determined tasks.

Thinking Aloud Protocol: During the course of a usability test, the test users are asked to verbalize their thoughts, feelings, and opinions while interacting with the system. It is very useful in capturing a wide range of cognitive activities. Two variations of thinking-aloud protocol technique are:
Critical response This requires the user to be vocal only during the execution of certain predetermined subtasks.
Periodic report This is used when the task is complex and makes it difficult for users to think aloud while performing the task at the same time. The user, therefore, verbalizes at predetermined intervals of time and describes what he is currently trying to achieve. The length of the interval depends upon the complexity of the task.

Usability Inspection

In Usability Inspection approach, usability specialists – and sometimes software developers, users and other professionals – examine usability-related aspects of a user interface. Commonly used Inspection methods are [C. Wharton]:

Cognitive Walkthroughs: Cognitive walkthrough involves one or a group of evaluators inspecting a user interface by going through a set of tasks and evaluate its understandability and ease of learning. The user interface is often presented in the form of a paper mock-up or a working prototype, but it can also be a fully developed interface. The input to the walkthrough also includes the user profile, especially the users' knowledge of the task domain and of the interface, and the task cases.

Feature Inspection: This inspection technique focuses on the feature set of a product. The inspectors are usually given use cases with the end result to be obtained from the use of the product. Each feature is analyzed for its availability, understandability, and other aspects of usability. For example, a common user scenario for the use of a word processor is to produce a letter. The features that would be used include entering text, formatting text, spell-checking, saving the text to a file, and printing the letter. Each set of features used to produce the required output (a letter) is analyzed for its availability, understandability, and general usefulness.

Heuristic Evaluation: A heuristic is a guideline or general principle or rule of thumb that can guide a design decision or be used to critique a decision that has already been made. Heuristic evaluation, developed by Jakob Nielsen and Rolf Molich, is a method for structuring the critique of a system using a set of relatively simple and general heuristics.
The general idea behind heuristic evaluation is that several evaluators independently evaluate a system to come up with potential usability problems. It is important that there be several of these evaluators and that the evaluations be done independently. Nielsen's experience indicates that around 5 evaluators usually results in about 75% of the overall usability problems being discovered.

Pluralistic Walkthrough: A group of users, developers, and human factors engineers meet together to step through a set of tasks, discussing and evaluating the usability of a system. Group walkthroughs have the advantage of providing a diverse range of skills and perspectives to bear on usability problems. As with any inspection, the more people looking for problems, the higher the probability of finding problems is. Also, the interaction between the team during the walkthrough helps to resolve usability issues faster.

Brainstorming: A method for generating ideas, intended to inspire the free-flowing sharing of thoughts of an individual or a group of people, typically while withholding criticism in order to promote uninhibited thinking.

Inquiry

Here, usability evaluators obtain information about users' likes, dislikes, needs, and understanding of the system prototype by talking to them, observing them using the system prototype in real work (not for the purpose of usability testing), or letting them answer questions verbally or in written form. Inquiry methods include:

Field Observation: Human factors engineers go to representative users' workplace and observe them work, to understand how the users are using the system to accomplish their tasks and what kind of mental model the users have about the system.

Focus Groups: This is a data collecting technique where about 6 to 9 users are brought together to discuss issues relating to the system. A human factors engineer plays the role of a moderator, who needs to prepare the list of issues to be discussed beforehand and seek to gather the needed information from the discussion. This can capture spontaneous user reactions and ideas that evolve in the dynamic group process.

Interviews: In this technique, human factors engineers formulate questions about the product based on the kind of issues of interest. Then they interview representative users to ask them these questions in order to gather information desired. It is good at obtaining detailed information as well as information that can only be obtained from the interactive process between the interviewer and the user.
In an evaluation interview, an interviewer reads the questions to the user, the user replies verbally, and the interviewer records those responses. The methods of interviewing include unstructured interviewing and structured interviewing.

Unstructured interviewing methods are used during the earlier stages of usability evaluation. The objective of the investigator at this stage is to gather as much information as possible concerning the user's experience. The interviewer does not have a well-defined agenda and is not concerned with any specific aspects of the system. The primary objective is to obtain information on procedures adopted by users and on their expectations of the system.
Structured interviewing has a specific, predetermined agenda with specific questions to guide and direct the interview. Structured interviewing is more of an interrogation than unstructured interviewing, which is closer to a conversation.

Logging Actual Use: Logging involves having the computer automatically collecting statistics about the detailed use of the system. It is useful because it shows how users perform their actual work and because it is easy to automatically collect data from a large number of users working under different circumstances. Typically, an interface log will contain statistics about the frequency with which each user has used each feature in the program and the frequency with which various events of interest (such as error messages) have occurred. Statistics showing the frequency of use of commands and other system features can be used to optimize frequently used features and to identify the features that are rarely used or not used. Statistics showing the frequency of various error situations and the use of online help can be used to improve the usability of future releases of the system by redesigning the features causing the most errors and most access for online help.

Proactive Field Study: Before designing a system, in order to understand the users, their tasks, and their working environment, human factors engineers go to representative users' workplace and talk to them, observe them work, and ask them questions, to understand the user characteristics, the work flow, the system features they need, etc. This technique should be used during the requirement or early design stage of software development. This should be the first step of usability work for a project.

Questionnaires: They are widely used instruments for usability evaluation, but their correct construction is often a complex task since several previous administrations are required to obtain a fine-tuned version of the questionnaire.

Chapter 4. *Task Analysis*

4.1 The task model: analysis of current and future situation

4.1.1 Root Concept

Before going into the field, an understanding of project's high-level goals is going to be developed. In SBD this understanding is documented as a root concept. See Table Root Concept.

Statement of project vision and rationale

There already exists a working system used by many companies. PersoneelOnline is the online solution for a company personnel and salary administration. It is a completely online personnel system but it needs some improvements to increase its functionalities and usability to make better user's experience. This new application that will replace the first one and which includes most of the required functionality is being developed. However its interface is not ready to handle the final users yet.

Component	*Contributions to the Root Concept*
High-level vision	*PersoneelOnline* is the online solution for a company personnel and salary administration. It is a completely online personnel system.
Basic rationale	A new application that will replace the current one and which includes most of the required functionality is being developed. However its interface is not ready to handle the final users yet. The technical parts are almost developed but without taking into account the layout and usability of its interface. This project develops the interface for a web-based management application following a user-centered design approach and usability evaluations.
Stakeholder group	
Payroll Specialist at Personeel en Zo	• *Background*: The Payroll Specialists at *Personeel en Zo* are mainly working with the Cobra system, which as it was mentioned on the Introduction Chapter, is an offline software tool of another company. • *Expectations:* They will use the system to make all the pay slip calculations, set everything about a new company, establish the social security settings and other things as make corrections and pay slips proformas.
Payroll Specialist of clients	• *Background*: The Payroll Specialists of clients are accountants with *Know-how* of payroll. They are only working with the PZ Web system. And they have to handle with multiple clients. • *Expectations:* They will use the system to make all the pay slip calculations.
Normal User	• *Background*: The normal users are payroll administrators but with no knowledge of payroll. They are working with PZ Web, using entire system. And they use to handle 1 to 3 companies with a maximum of

	50 employees per company. • *Expectations:* They want to use the system for entering new employees, entering loon components, receiving signals and they like to work with wizards.
Payroll Specialist of one client	• *Background*: This group is for the ones who are payroll specialist of one client but with more than 50 employees. They just introduce information to build the pay slips. They never get closer to a company and don't know what is going on them. The pay slip generation can be resumed into three steps: Introduce the information. Make a request to get the pay slip processed and print the pay slip. They are also the ones where people go to complain about mistakes on their pay slips.
Human Resource Specialist of clients	• *Background*: They are Human Resource specialist. They have to handle with only one company with more than 50 employees.
Employee	• *Background*: This group is formed by the employees of a company. They don't necessary have to have *Know-how* about payroll or HR. • *Expectations:* They will use the system for seeing the pay slips, adding signals, changing personal information and entering some other information as worked hours.
Debtor	• *Background*: A debtor is everyone who owes money. • *Expectations:* They will use the system to see the invoices, see where the invoices are going; see an invoice history and get invoices by post between others.
Starting assumptions	The new system is based in the current Cobra and PZ Web applications. The implementation of the system engine is very advanced.

Table: Root Concept

4.1.2 Analysis of current practice

Task Analysis

After having several talking session with the client and the future application users, the different roles a user can play on this application were found as well as the different user levels and the tasks a user can carry out depending on his user level and role. To see the tasks and roles in detail, see Appendix A.1

Interviews

In order to get a feedback about the system from the real users different guides have been developed for each stakeholder. With these guides it is going to be known the kind of users they are, the tasks they work with, and overall, which features of the system they are using, they like the most and which ones they will like to improve, change or add in the new system.
Most of the questions are quite the same as they will be working in the future with the same system. But this is also very useful so the different answers that different kinds of user give to the same question can be compared.

The interviews can be thought of as a "conversation with a purpose" [Kahn and Cannell, 1957]. The interviews will be written with open-ended questions as the goal is to gain first impressions about how users work with the current system and what users expect for the new one. These interviews are semi-structured interviews. A basic script for guidance will be followed so that the same topics are covered with each interviewee. Starting with preplanned questions the interviewee will be probed to say more until no new information was forthcoming. There are two types of interviews, with most of the question in common but some others different depending on if the user type. These interviews are available on the Appendix A.2 and just to let the reader have a brief idea about how they are, here the main blocks of questions are enumerated:

1. Open-ended questions.
2. Talking about Cobra system.
3. Talking about PZ Web system.
4. Conduct task analysis and prioritize tasks.
5. Measurable usability objectives.
6. Expectations, requirements and preferences regarding the new PersoneelOnline.
7. Any other comments.

4.1.3 Summarizing the Field Data

Interviews outcomes

The observations and interviews for each stakeholder group are organized into stakeholders´ profiles. These profiles summarize the general characteristics of each group, and are based on the observations and interviews above.

The interviews are semi-structured interviews. Because of that maybe some questions were not covered or even answered in any other question as the result of the conversations that came up when talking about each topic.

The table Interviews Outcomes shows the main ideas of the interviews' outcomes. It is structured on the following way: on the rows are the different blocks of questions, and each column is a different stakeholder. This way is very easy to compare the different answers each stakeholder gave to the same question.
To see the interviews' outcomes go to Appendix A.3.

	Payroll Specialist at Personeel en Zo	Payroll Specialist of clients	Normal User	Payroll Specialist of one client	Human Resource Specialist of clients	Employee
Open-ended questions	- Payroll administrator. - Currently working with Cobra.	- Payroll specialist. - Currently working with PZ web.	- Grew up with computers and use it every day. - Use the computer to get all kinds of information and to communicate.	- Book keepers. - Office applications.	- They work as users and administrators.	
Talking about Cobra system	- They use it: * To make payslip's calculations, * set new companies, * establish the social security settings, * make corrections, proformas ... - They would like it to be more friendly and trustable.					
Talking about PZ Web system	- They know how PZ Web works. - They have to process and answer the client requests.	- They use this application to provide the payroll administration to their customers.	- They use it to provide their employees the pay slips.	- They use it for processing pay slips.	- They find it very easy to work with and friendly.	
Conduct task analysis and prioritize tasks	- They use the system to: * Implement new clients. * Processing the salary. * Make changes in the settings. * Make analysis. * Download reports in Excel and make overviews. * Make offers for a company. * Put back the payroll period. - Critical tasks are: * The deadline of the payroll administration. * Speed of the system. - The most important for them is to process payslips.	- The feature on the system they use the most is managing looncomponents. - Critical tasks are: * Immediately payroll process. * Make fewer mistakes. - The most important for them is the payroll processing system and to be able to make proformas.	- They use the system: * To have some insight to all the different employee's information. * As a way of communication with their employees. - Critical tasks are: * To pay the employees on time. * To give to the employees a correct payslip. * Everything correct to the tax company. * They need to work in a quick and handy way. - Features they use the most are: * Quick possibility to insert	- They use the system to: * Enter payroll processing mutations. * Fill in loon components. * Add and dismiss employees. - Critical task is the Standard Register. - Very important are the tasks of entering looncomponents and making fast calculations.	- They use the system to: * Make contracts. * Make the EDM. * Salary changes. * Creating an employee. * The Verlof and Verzuim tabs.	

		the monthly mutations. * To ask for a salary calculation. * To see the monthly results. - Tasks to accomplish easily with few errors: * Hanging a result of a meeting or a contract. * Vacation questionnaires.			
Expectations regarding the new PersoneelOnline	- Save button always visible. - Don't have to type the same information several times. - Group tasks. - To solve the problem in the updates. - To have a list in which you can see when a company is already implemented. - To have a Data Management System.	- Be able to create/modify companies/employees. - Be able to make proformas. - Process the pay slips themselves. - Give their clients access to the system and the feeling that it is their system. - To see what a customer has done on the system. - Know the result of what they are doing immediately. - Manage signals. - Access from every location at every time. - Easy to use. - Speed in processing.	- Chance of asking the system for support at any time. - Information good organized. - They would hide all the irrelevant information. - To give to their employees his own digital pay slip. - Access from every location at every time. - Speed in processing.	- Save button always visible. - To have the import function. - To be able to make Proformas. - To have an overview about all the changes made by someone else.	- Save button always visible. - Overviews. - To have "Suggestions" and check lists. - Previous run so they can check if everything is going ok. - Employee photo. - Save several things at the same time. - Possibility of checking what you have changed or done. - Improve the management of an out of order employee.	- See/Ask for holidays (Verlof) - See news. - Handbook. - See and fill in some kind of document. - See/change his own personal information. - Hour registration. - See their own pay slip online.

Table: Interviews Outcomes

Chapter 5. *Prototyping*

5.1 Overview

This chapter is divided on three sections: First, Second and Third Prototype. These prototypes are made in an iterative way as each prototype is first developed to later on make an evaluation over it and start with the following prototype.

When this project starts some studies about how the interface should be, are already made. There is what can be called a Prototype Zero, make on Excel files in which each file represents a page of the new application.
It is a really good starting point for the project as it gives a brief idea about what the aim of this application is and it becomes the base up on which the First Prototype is built.
Because it is very difficult to make a quick and easy revision and layout of pages, an easy way to demonstrate the prototype to clients and team members should be found. The idea is to make these Excel files as HTML pages so it is easier to go from one page to another using links and other HTML controls. At this point is when the First Prototype comes up.

After studying the user needs and application requirements the First Prototype is not good enough. It is not easy to navigate and it is difficult to group information in similar screens. So with the Second Prototype is built what in the Background Chapter was mentioned as *mockup* or *horizontal prototype.* Here all the application features, although without going into its implementation, is explained.

The Third Prototype is a *proof-of-concept prototype* or *vertical prototype* in which the main features of the application through each layer of the design are built out. This one supposes the final result of the project as it covers all the interface elements and application requirements making it possible to use it as a demonstration for users. If the new product is compared with the applications it is based on, it is more users friendly and it works more efficiently and faster, which are the goals for this project.

5.2 First Prototype

5.2.1 Tool used

This First Prototype is completely based on some Excel files the people working on this project used to have to handle everything about the interface, navigation and presentation layer of the new web based salary application.
Some prototyping tools are visual or text based HTML tools and diagramming tools.
In June 2002 GUUUI [5]conducted a survey on web prototyping tools usage. The purpose of the survey was to find out what tools are used for prototyping, what requirements interaction designers have for their tools, and how happy they are with the tools they are using. Some key findings on this survey are:

- There is currently no distinct consensus on what is the best tool to use for web prototyping.
- HTML and diagramming tools are the most commonly type of tools used for web prototyping.
- Macromedia Dreamweaver and Microsoft Visio are the most widely used tools.
- Respondents feel that the tools they use fall short on some high priority criteria, such as lying out and making changes to pages.
- Few seem to be perfectly happy with the tools they are using though users of HTML tools are strikingly more content than users of diagramming tools.

This first prototype is built in HTML because:
- It's quick and easy to layout pages. It will be very easy and quick to navigate through the application.
- It's quick and easy to make changes to pages.

[5] GUUUI is a site for people engaged in the various fields of making the web a better experience for the users.[GUUUI]

- It's quick and easy to do site wide revisions to the prototype.
- It's able to simulate basic functionality.
- It's easy to demonstrate the prototype to clients and team members.
- It's suitable for carrying out usability testing.
- It produces good looking prototype.
- It fits in with existing development tools. This is very important as in the end the application will be developed with ASP.NET.
- It's easy to learn and easy to use, so there is no need to spend too much time on building the prototype having more time to evaluate it.

This prototype is developed in Microsoft Visual Studio 2008, working only with the HTML part.

5.2.2 Navigation

The application navigation is mainly based on the way a business is organized. Inside a business there are different levels as Organization, Companies and Employees. And is on the same way how the application is organized.

In this new web based salary management application there are two different levels: Company and Employee level.
Based on the existing PZ Web used by Personeel en Zo BV, it has a left menu which will let to switch between both Company and Employee level by clicking on an employee or company name. Because it is a too complex functionality to develop on a First Prototype, two buttons, one for Employee level and another for Company level, are added to switch between those two levels.
In both Company and Employee level the following navigation structure is applied:

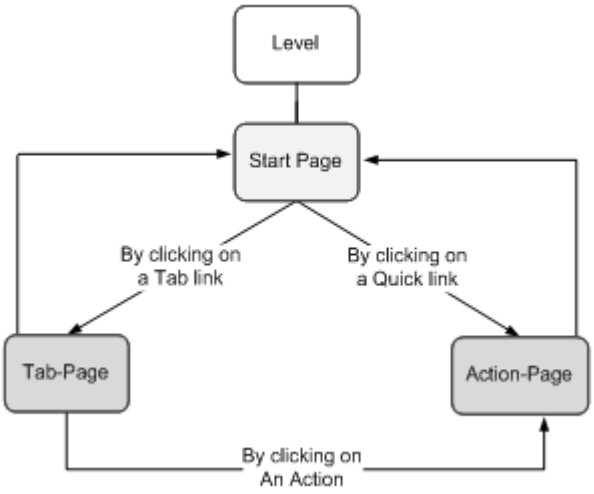


Figure 3: First prototype navigation diagram

There is a **Start Page** per Level. This starting page consists of:
- Employee/Company Photo/Logo.
- Employee/Company Information.
- Several links to Tab-Pages and Action-Pages.

Tab-Page is called to a page that group different kinds of information and functionalities. The following Tab-Pages can be found depending on the level:

Employee Level

- *Personalia (Personal Information)*
- *Dienstverband (Employee information)*
- *Loonelementen (Remunerations)*
- *Loonheffing (Levy)*
- *Personeelsdossier (Personal Documents)*
- *Loonstrook (Pay slips)*

Company Level

- *Salaris verwerking (Salary processing)*
- *Personeels zaken (Staff Management)*
- *Bedrijfsinstellingen (Company settings)*

Some of these Tab-Pages have buttons or links to Action-Pages.

Action-Page is the page in which certain actions can be carried out, actions as editing or including new information. These pages can be reached since the start page through a quick link or since a Tab-Page by clicking on a button or a link.

On the Appendix B.1 (Navigation) there are some screenshots showing how these types of pages look like.

5.2.3 Screen types

With *screen type* is denoted a screen layout that makes the user identify a certain type of page or popup. Just looking at it, the user must have an idea about what he is meant to do on that page without getting lost or confused.
As it is shown in the next section, *Evaluation*, the screen types are not very well established in this prototype.

Most of the pages consist of an editable form or table, already filled in with the default or current values. If the user wants to change the current information or even add new information he has to delete the information on the form, write the new one and save. Usually the *Save Button* is at the bottom of the page, sometimes hidden because the page is too large and the user has to scroll down to see it; and there is a table with a historic in most of the pages, with all the past information on it.

5.2.4 Evaluation

Technique used

The technique used for this prototype evaluation is Brainstorming. The client, the developer, an employee working for this particular project and the presentation layer developer meet together to discuss and evaluate this prototype.
This is the best way of evaluating this prototype as there are a lot of features to talk about and new ideas to discuss. As it is a prototype completely built in HTML it doesn't contain functionality enough to use any other type of evaluation method in which a group of tasks are needed to go through them.

Current situation

In the first prototype are missing screen types for the different panels.

The navigation through the application is made with links.

The way to add new information is not the best. Actually there are two ways:

- The current information is shown in a table. If the user wants to add new information he has to press the "new button" and is led to another panel. In this new panel there is a form already filled in with the current information. This information has to be changed in order to write the new one.

- The current information is in a form, having to change this information to add new one.

Problems

The navigation through the application is very difficult.

Due to the lack of screen types the site is messy and not uniform.

The most confusing is the fact that there are two different ways of showing the current information.

Another problem is the fact that the current information is shown in a form:

- This is confusing for the user. He doesn't know if the values on the form are default values; what happen if he changes these values? Is new information created or is the information removed forever?

- The forms take a lot of space in the screen and sometimes the user has to scroll down to see all the information. Due to this, the "save button" is not always visible and the user can forget about saving changes.

- Sometimes is not necessary to show so much information and several text boxes in the form are empty.

The last problem found is that the historic form is always visible when adding new information:

- Why to show always information that the user hasn't asked for?

- The system becomes slower because the historic has to be loaded every time.

Solutions

Discussing all these problems in a brainstorming way the following ideas improve the First prototype and are used to build the Second one:

- Find the different panel screen types to make the site more visual uniform and intuitive.

- Remove all the forms showing current information in a way that:
  - The current information is always fixed. Not in forms anymore. This new model supposes a big change in the First prototype. As the current information is not in forms anymore, it doesn't take too much space in the pages. The idea of building a start page in which the user can see an overview of all the current information is born at this point.

  - There is a page with all the possible overviews in a dashboard. There will be two buttons to edit/create and show details. And this page will be configurable as the user will be able to close or add dashlets in order to make it more personal and not see information that he doesn't want to see.

  - When editing/creating information a popup containing a form appears:
    - The form can have more information than the one showed in the overview of the current information.

    - The form appears already filled in with the current information or/and default values depending on if the user is editing or creating new information: Most of the times there is no need to change all the information but just some fields, so why should the user fill all the fields in again?

    - There are only two ways of closing the popup. The user is not able to click anywhere else outside the popup; so he has to click either Save or Cancel to close it. This way there are two buttons:
      - Save button, to save the changes and close the popup.

      - Cancel button, to cancel the operation and close the popup without saving.

    - The details button leads the user to another page that shows the details of the current information. Inside this page there is a button to go to the historic panel which appears as a popup. In this way the user can see the historic when he wants to. In this detail page there is the chance of editing/creating information the same way as mentioned above.

Conclusions

On the following table the prototype problems and solutions are showed. To see some screenshots regarding the first prototype problems and some comments about how they were solved go to the appendix B.1 (Evaluation).

Problems	Solutions
– Missing screen types. – No good way to add new information. – Historic always visible. – Difficult navigation. – Too much information in one page.	– Find screen types. – *Fix Current* information. – Customizable Dashboard as starting page with overviews. – Popups to edit or create information. – Easier to navigate and user friendly.

5.3 Second Prototype

5.3.1 Tool used

This Second Prototype is basically built using HTML for almost everything but some JavaScript functions. Some very simple ASP.NET functionalities as *Master* and *Content Pages* are used as well.
Most of the features are represented on this prototype, although the functionality is not there. There are a lot of possible clickable buttons and links but they are just HTML images with no functionality at all.
This way is represented what the user is able to do on the application, but as it is not the final prototype yet, it would be a waste of time going too much into technical details as some changes can still be made. This is a *horizontal prototype*.

5.3.2 Navigation

The Second Prototype is a completely new way of thinking. The diagram *Second Prototype Navigation Tree* shows the navigation tree that came up after the First Prototype evaluation. It seems much more complex, but is not. Everything is well defined and placed and the navigation through the application is more intuitive.

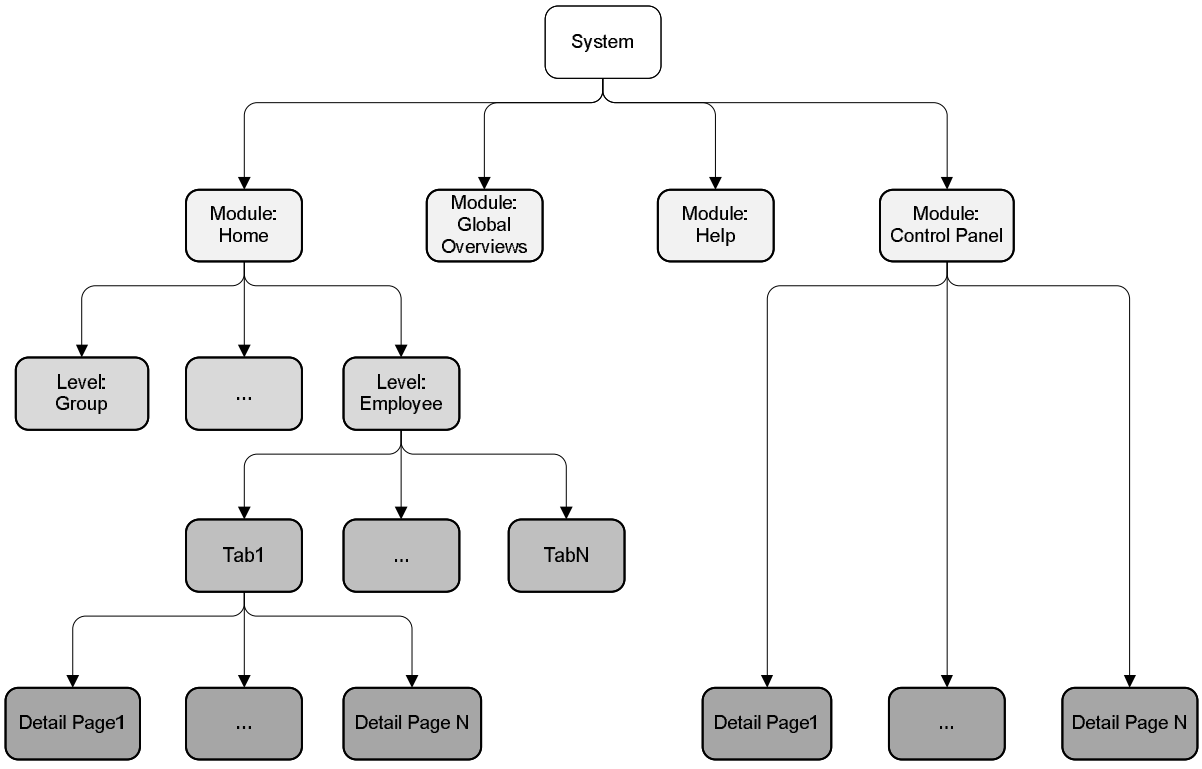


Figure 4: Second prototype navigation diagram

The system is divided into four different modules or top-level pages: *Home*, *Global*, *Control Panel* and *Help*[6].
Home: This module represents the application itself, where all the calculations are done. It is divided into four levels: *Group*, *Organization*, *Company* and *Employee Level*. Each user belongs to a level depending on his user type. And it is able to manage all the levels below his own level including his own level. For this the following hierarchy must be taken into account:

[6] These are the names given to the different modules just to call them in some way. But maybe they won't be the final ones. It is not so important at this point to worry about these small details.

Figure 5: Levels hierarchy

For example, if an organization user gets into this application, he will be able to manage his own organization, all the companies that belong to his organization, and all the employees belonging to the companies of that organization. But he could never manage or even see anything at Group Level.

Global Overviews: This module gives us the possibility of having different types of overviews for the organizations, companies or employees without having to select one in particular.
For this prototype this module is not developed in detail.

Control Panel: Here take place all the different functions and information regarding settings and system configuration.
The layout and appearance is very similar to the Windows' control panel. This is done on purpose as its aim is very similar to the Windows´ control panel and on this way the user feels more comfortable because it's something familiar.
The information is organized in three columns: The first column has to do with salary settings, i.e. SVW Settings, Sector, Risk Group, CAO Salary table, Loonmodel, Sattelite Data, Personnel Options and Engene Log. The second one has to do with uses roles, profiles and accounts and the third column is for clients, groups, organizations, companies and employees management.

Help: Here is where the help functionality is placed. This module is not developed in detail either.

To see a *Navigation Diagram* that explain more visually how to navigate through the application go to Appendix B.2 (Navigation).

5.3.3 Screen types

A very important task to be developed in this prototype was to find screen types for the pages, so the application was more uniform and easy to learn and navigate.

Four different screen types for the details pages and three for the popups have been found. Although they are explained on detail on Appendix B.2 (Screen types), basically the detail pages are pages in which the information is fixed and in which there are different buttons to carry out different actions normally over popup window.
And about the popups, everyone consists of a form or a table with editable information.
For this application, modal popups are going to be used. When a user opened a popup, he couldn't click anywhere else but inside the popup. As it was analyzed on the user interview outcomes, a very common problem is that the users forget about saving changes quite a lot of times. With these popups, the user is never going to forget, as to close the popup he has to click either on the Close or Save button.

5.3.4 Interface elements

In this section it is described in detail each of the interface elements and when and why they are used.

Header

The header is going to place both static and dynamic information. Static information because there are some elements as the application logo, the date and time, user name and the logout link, that are always visible. And dynamic because there are some other elements that could change depending on the user level or the current selection: employee photo, company logo, employee and company name, period selector and system mode. After thinking what will be the best layout the following header came up.

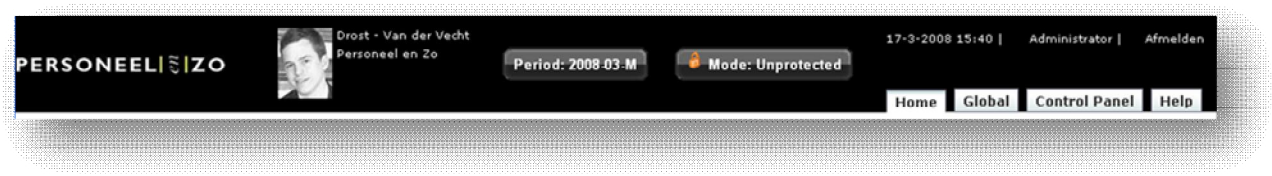


Figure 6: Header layout

Application logo: the application under developed will be sold to third parties, and they would be able to add his own logo on the left part of the header, making it much more personal in a way they can think they are using their own application.

Employee photo and **Company logo**: The main idea is to show over here the selected employee photo or the selected company logo so the user can always have in mind with whom he is working at every moment.

Employee and **Company name**: As it is for the employee photo and company logo, the user can see on the header the selected employee and/or company name.

Period selector: Most of the calculations made on this application have to take into account the period in which they are applied. A period consists of a *year*, a *period type* and a *period* belonging to that period type. There are three types of periods: monthly (M): 12 periods, weekly (W): 52 or 53 periods depending on the year and every four weeks (4W): 13 or 14 periods depending on the year.
With this *period selector* the user can change the period into the past, the present or the future in order to change/see that period information.

System mode: This is a very important feature on the application. It can be *Protected* or *Unprotected.* Some kinds of users are able to switch between protected and unprotected. If the system mode is *protected*, the user is not able to use some operations that are only allowed on *unprotected* mode. This is to protect the user against making changes that could affect the calculations already made. With the period selector, the user can go into the past. If he wants to change something over there it could affect calculations that have been already processed. If the system is on *protected* mode, those changes won't be possible and the calculations are not modified. But if the user is on *unprotected* mode, he is allowed to make changes that can have side effects on past calculations, and it will be always under the users' own risk.

Date and time: It shows the current date and time.

User name: It shows the current user nick-name. By clicking on it the user could access to his profile in order to change or see his profile information.

Log link: To log out.

All these elements have to appear on the header as they are common on every page, and it is important for the user to have them always visible.

Left Menu

In this new prototype the left menu mentioned in the first prototype is included. And it appears when the user is on the *Home Module.* This left menu contains tree panels:

- **Organization panel**. It shows all the organizations a group user is allowed to see. In case it was an organization user, it shows the organization the user belongs to. On Company and Employee Level this panel doesn´t even appear on the system.

- **Company panel**. It shows all the companies a group or organization user is allowed to see. In case it was a company user, only the company the user belongs to is shown. On Employee Level this panel doesn´t even appear on the system.

- **Employee panel**. It shows all the employees a group, organization or company user is allowed to see. In case it was an employee user, its own information is the only one shown in this panel.

These three panels work as filters as if the user makes a selection in one of them, the panels below are filtered by that selection. For example, if the user selects a particular organization, the company and employee panels will be filtered showing only the companies and employees that belong to that organization.
For this prototype it is known to work on this way, but it is on the Third Prototype where this functionality is implemented.

The user is able to collapse the panels if he wants to.

Each panel has a text box to look for the names in an easy way.

The user will be able to sort each panel list by name or number.

There are three buttons on each panel: *New, Management* and *Filter* button:
- **New button**. This button is a quick link to create a new Organization/Company/Employee.

- **Management button**. It is another quick link. This time to open the management page for the Organizations/Companies/Employees.

- **Filter button**. By clicking on this button a small popup with a check list appears and the user is able to select which filters he wants to apply to filter the panel list. An examples of filter could be, filter by department.

This left menu is mostly used to switch between levels. By clicking on an organization, company or employee name the user can see information regarding that selection.

There is almost one panel per level, although client and group levels are not going to be included.
Client panel is not in the application because it is not possible to switch between clients; there is always only one, and there is no information to show about a client.
Group panel: At first, it was thought to show all the groups a client user is allowed to see. In case it was a group user, it will show the group the user belongs to. And on Organization, Company and Employee Level this panel won't even appear on the system. But after thinking more deeply about if it was really necessary, it was found not to be useful. There are no overviews per group, so this panel is not needed on the left menu because there is no information to show.

Tabs per Level

Each level has at the most three different tabs. This is a very important improvement on the application user interface, as the one on the systems this one is based on, used to have several tabs what made the site very difficult to navigate.

On Company Level there are three tabs:
- **Company Information Tab Page**: This tab page consists of a dashboard with several dashlets that the user can move around or even close in order to customize his own application interface. In this tab page all the dashlets have to do with information related to the company selected. Information as company contact person, signals, salary details…

– **Settings Tab Page**: In the same way as the Company Information Tab Page it consists of another dashboard but this time its dashlets contain information about the company settings: default schedule, salary settings, etc.

– **Overviews Tab Page**: Here all the company overviews are shown using different kinds of chart representations as pie, column or area charts.

There are three tabs on Employee Level as well:

– **Personal Information Tab Page**: It is like the Company Information Tab Page but all the information now has to do with the employee selected: Employee schedule, absences, if he has a lease car, etc.

– **Pay slips Tab Page**: The idea of this tab is that the user could see or even print his pay slips.

– **Overviews Tab Page**: Employee overviews are shown.

Dashlets and Detail Pages

Dashlets are the main feature on this Second Prototype. As it was discussed during the First Prototype evaluation, on this prototype the current information is not going to be shown in editable forms or tables anymore, but is going to be fixed, so much less space is needed to show the same information. Every dashlet contains an overview with the most important information. And this information is shown in detail on that dashlet´s detail page when the user clicks on the details icon placed on the dashlet.

A problem was found on the detail pages: There is quite a lot of information per detail page. When saving changes it is too much information to be saved. For this reason different areas are made within one detail page. And to keep the consistence with the dashlets, they are also divided in different areas where each dashlet area contains a detail page area overview.

Usually every dashlet has a default action triggered when clicking on a dashlet's area. And this action is the same as the default action on its detail page area, usually edit.
As in the dashlets there is only that default action, is in the detail pages where all the functionality takes place; functionality as Edit, make New or show the Historic in general.

Some technical features

Two technical decisions have been already made for this second prototype:
Retractable Left Menu: Users need to access the navigation but they may need the screen space even more. For this reason it is created a menu that can be put aside and easily retrieved again by just putting the mouse over. This is implemented using dynamic html, which also requires some form of Liquid Layout[7]. This pattern mainly saves a lot of screen-space while having the menu still available.

Make the main content scrolling: Some times, because there is a huge amount of information or because the user resizes the window, some information is not readable because is not visible. So the user has to be able to scroll in order to read this information. This is made in a way that only the main content can be scrolled (i.e. everything but the header, left menu and tabs). This is very important because in this way the left panel, header and tabs are never out of sight and the user never get lost.

On the Appendix B.2 (Technical features) some screenshots are shown.

[7] The basic page design must be such that text is readable already. But when users enlarge their browser window, or put a menu aside the readability must increase. We must decide which parts of the page will scale along and which ones stay fixed. There is always a main content area that is on the Center Stage. That one should scale. Other columns with related stuff or navigation usually do not scale.

5.3.5 Evaluation

Technique used

Basically there is not an evaluation phase for the second prototype but it is done during the development of the third prototype; this prototype covers all the features wanted to be on the application and the third one is started making the second prototype evaluation at the same time. It is very important to take into account that this project is being developed in the same place together with the client and users, so the evaluation is continuously made on each new task.

Problems

Starting with the header, the following issues are found:

- On one hand it is very useful to see the employee photo or company logo always on the header, so just looking at it the user knows exactly with whom he is working. But on the other hand, it is not good to have so much dynamic information on the header, as consistency in design helps users to navigate through the site efficiently without being distracted by unexpected changes.

- The design of the period and mode selector is not the best. Why are they more visible than the employee or company names? Are they more important? Why labels are needed to tag them?

Dashlets:

- The dashlets design is not the best. They will be draggable but there is not a defined place on them where the user can place the mouse over when he wants to drag the dashlet.
  The user must also be able to collapse or close a dashlet, but the place where those icons are placed and the way in which a closed dashlet is going to be added again on the user dashboard is not defined yet.

- Another problem found in the dashlets is the difficulty of reaching the icon on a dashlet to go into the detail page. This is a very common action and it should be very easy to execute for the user.

On the left menu there are the following problems:

- The most important is the fact that the panels' height is static. It doesn´t matter the amount of items a panel has, it height is always the same. This way it can be a panel with only one item with blank space on it and a panel with a big amount of items in which the user has to use the scroll bar to see them all, where it would be better if the second panel takes advantage of the blank space of the first one to not have to use the scroll bar.

Talking about the historic functionality:

- As it is explained on the Appendix B.2 (Screen Type d2), the delete button on a detail page removes the detail page information, showing the previews one. And actually this is the problem. It was meant to delete the first row of the historic, and as it is possible to change information in the past or in the future, the information shown on the detail page might not be the one that the user wants to remove (which is the last one he created). This will make the user get confuse.

Solutions

During the third prototype development the following solutions to the problems mentioned above were founded:

Talking about the header:

- The employee photo/company logo is removed and only the employee and company names stay. On this way the header is more consistent and also becomes smaller, giving more place to the application information itself, which is much more important. As these elements are still important, they are included into the *Personal/Company Information* dashlet.

- The label for the Period is removed. A user of this application will know what it is about just with the Period information.
  The same with the System Mode label. To use a locked/unlocked icon is enough to represent the protected/unprotected mode.

Dashlets:

- A header is added on every dashlet which emphasizes more what is the dashlet about and also defines a place which the user can use to drag the dashlet around. On this header the close button and the one to collapse the dashlet content take place.

- The detail icon on the top of the dashlet is replaced by a bar at the bottom of the dashlet. As it is larger, bigger and as a result more visible, it is much more user friendly and makes the application much easier to navigate.

- A catalog to replace closed dashlets is included on the application. An icon with a plus and a tooltip, *add dashlet*, is placed on the top right corner of the main content area. When the user clicks on the icon a panel with only the necessary size is opened. This way as many possible dashlets there was bigger the panel will be. As it is a panel which overlaps the main content and it is on the right side, it doesn't disturb the user at all.
  Only the dashlets which are not on the dashboard appear on the catalog. And just by clicking on a dashlet name, it will appear on the dashboard and disappear from the catalog.

Left menu:

- The panels on the left menu are resizable. According to the items they have and the other panel items, the panel gets more or less space, taking the size of the screen also into account. If a panel has too many items, and its size is not big enough to show all of them at once, a scroll bar appears.

Historic:

- The remove functionality is on the first row of the table which appears on the historic popup. This way the user always knows which information is deleted.

Conclusions

The following table shows the problems and solutions of this prototype. On the Appendix B.2 (Evaluation) can be seen how some decision solving problems were taken and some screenshots regarding the changes are shown.

Problems	Solutions
– Header too dynamic. – No structured dashlets and lack of catalog to add new dashlets. – Panels on the left menu are static. – Delete function on the Historic is not clear enough.	– Make header more consistent. – Dashlets well organized and with a catalog to add. – Panels on the left menu resizable. – Delete function on the Historic is well defined. The last change made is deleted.

5.4 Third Prototype

5.4.1 Tool used

The third prototype is a *proof of concept prototype* or *vertical prototype*. It demonstrates a working, though incomplete, system for key functions. The ASP.NET framework, JavaScript functions on the client side, C# on the server one and AJAX, are used on the implementation of this prototype.

5.4.2 Interface elements

This prototype goes vertically into each of the interface elements developed for the Second prototype. The functionality of the second prototype components is implemented.

Header

It is already known how the header layout is like, but now it is time to make it work.

The area in which the selected period is show is clickable in a way that a popup appears and the user is able to change the period.
If the user changes the system period either into the past or the future, the selected period shown on the header gets a different color and becomes bigger, so the user can notice very easily he is not in the current period and all the changes he made can be dangerous.

Figure 7: Past Period

Figure 8: Present Period

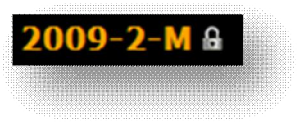


Figure 9: Future Period

Something similar happens with the *Mode Selector*. Just by clicking on the lock image the user changes the mode from protected to unprotect and vice versa. As to change from protected to unprotect mode is in some way dangerous, the user must confirm his decision with a *Confirm Dialog* before getting the mode changed.

Left menu

The Second Prototype explained some functionality which is implemented in this prototype:

- Filter a panel´s content making a selection on a panel above.
- Filter a panel´s content using the Filter button. By clicking on it a popup appears in which the user can choose the filters to apply.
- Filter a panel`s content each time that a character is written on the text box that appears on each panel. If a selection is already made on that panel, the selected item is not affected by this kind of filter.
- The items on each panel can be sort by name or by number.

For this prototype a new feature is included, *resizable panels*. The application must use as many space as it can on the user screen and the panels size must be proportional to the amount of items each panel has.
To implement this functionality, the following must be taken into account:

- The space that the left menu occupies on the screen.
- The items of each panel.

This formula is used to calculate the size of each panel every time a panel changes its amount of items:

$$panelSize = \left(\frac{panelItems}{totalItems}\right) * (leftMenuSize - (noItemsInfoSize))$$

Where:
panelItems: number of items a panel has.
totalItems: the sum of the items of all the panels.
leftMenuSize: the size that the left menu occupies on the screen. It is calculated with the *offsetHeight* property for an element, in this case for the left menu div.
noItemsInfoSize: the size of all the elements on the left menu that are no panel items.

Dashlets

As this project has been developed under the ASP.NET framework, the existing Web Parts were used at the beginning. A web part is an ASP.NET server control which is added to a Web Part Zone on Web Part Pages by users at run time. Web Parts are an integrated set of controls used to create Web sites that enable users to modify the content, look, and functionality of Web pages directly from the browser. The changes are then saved for the user and recalled for subsequent visits.
Some advantages of using Web Parts are:

- The Visual Studio.NET drag and drop creation and configuration of Web Parts in the visual designer, is a great boon. It improves speed of development.
- Web Parts provide users an easy way to customize the website at runtime.
- Any existing ASP.NET control can be used as a Web Part control.
- There is no need to reinvent the wheel.

But using this ASP.NET Web Parts, there are some problems and limitations:

- Drag and drop facilities are not available for some browsers as Firefox or IE7.
- It works mainly with Post backs[8], which sometimes are difficult to handle and make the application slower.
- Not all the functionalities that the Web Parts provide are needed. And some functionality that is not provided may be.

Due to these problems a special dashlet for this project is built without using the ASP.NET Web Parts. They work mostly on client side using AJAX (Asynchronous JavaScript And XML) that enables the Web application to increase interactivity, speed, and usability.

Popups

A popup is a graphical user interface (GUI) display area, usually a small window that suddenly appears ("pops up") in the foreground of the visual interface. Popups can be initiated by a single or double mouse click or rollover, and also possibly by voice command or can simply be timed to occur. A popup window must be smaller than the background window or interface; otherwise, it's a replacement interface.

Two different kinds of popups can be defined, the *normal* popups and the *Ajax Toolkit Modal Popup*.

A *normal popup* is just a child window for a parent application. Interaction with the rest of the page is always possible.
With JavaScript they are very easy to implement as it is only needed to use the *Open Method* which opens a new window and loads the document specified by a given URL:

NewWindow = window.open([strURL] [, strName] [, strFeatures] [, boolReplace])

Where:
strURL: String that specifies the URL of the document to display.

strName: String that specifies the name of the window.

[8] A Post back is an action taken by an interactive webpage, when the entire page and its contents are sent to the server for processing some information and then, the server post the same page back to the browser

strFeatures: String that contains a list of items separated by commas. The following features are supported: *channelmode, directories, fullscreen, height, leftlocation, menubar, resizable, scrollbars, status, titlebar, toolbar, top, width and zoominherit.*

boolReplace: Boolean that specifies whether the sURL creates a new entry or replaces the current entry in the window's history list. This parameter only takes effect if the sURL is loaded into the same window.

The *Ajax Toolkit Modal Popup* extender allows a page to display content to the user in a "modal" manner which prevents the user from interacting with the rest of the page. The modal content can be any hierarchy of controls and is displayed above a background that can have a custom style applied to it. When displayed, only the modal content can be interacted with; clicking on the rest of the page does nothing. When the user is done interacting with the modal content, a click of an OK/Cancel control dismisses the modal content and optionally runs custom script. The custom script will typically be used to apply whatever changes were made while the modal mode was active. If a post back is required, simply allow the OK/Cancel control to post back and the page to re-render. A modal popup can be also absolutely position by setting the X and Y properties. By default it is centered on the page, however if just X or Y is specified then it is centered vertically or horizontally.

In this salary web based application are only used Modal Popups. The fact that they prevent the user from interacting with the rest of the page is the most important reason why this kind of popups is used in the application. With this kind of popup the user will never forget about saving changes as he has to click either Save/OK or Cancel to close the popup window and keep on working on the application. Some other advantages of using the Ajax Toolkit Modal Popup extender are:

- It is easy to use.
- Allows dynamic popups (possible to combine with update panel)
- Supports server controls.

But there is also a big disadvantage of using this kind of popups on this application:

- Window content must be already in the code to be possible to open the popup without an extra server roundtrip. To allow multiple popups in a web page to open without roundtrip, all code must be there. This application is mostly based on popups and there are going to be more than fifteen popups per page. If these Ajax Toolkit Modal Popups are used the pages will be too heavy.

It is used a new popup window implementation which solves the problem of having the code of every popup already load on the page and keep all the advantages of the Ajax Toolkit Modal popup. Its main properties are:

- The code is loaded dynamically.
- Allows complex user interaction (not just static text).
- Is easy to use.

Period Picker

This application is basically based on periods. The user will have to run it for a period and each company can have a different period type: monthly (M), weekly (W) or every four weeks (4W).
For this reason a component to choose the period for each period type is needed. In the old applications it was just composed for four dropdowns, one per period type and another one for the year; but for this new application a component which helps the user and requires less clicks to set the period is used:

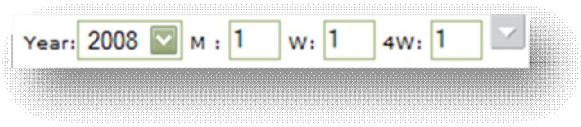


Figure 10: Period picker

It consists of two dropdowns, one for the year and another for the periods.
When the periods´ dropdown is expanded an area with three columns appears:

- Each column has the period values for each period type depending on the year selected. It is shown in a way that it is easier to know which periods for the different period types belong to the same time slot. For example, the moth 1 has the weeks 1, 2, 3 and 4 and the four weeks 1; and all this information is on the same row.

– On the bottom is shown the start and end date of the period that was just selected.
– If the period selected don´t overlap on time, and icon is shown warning it.

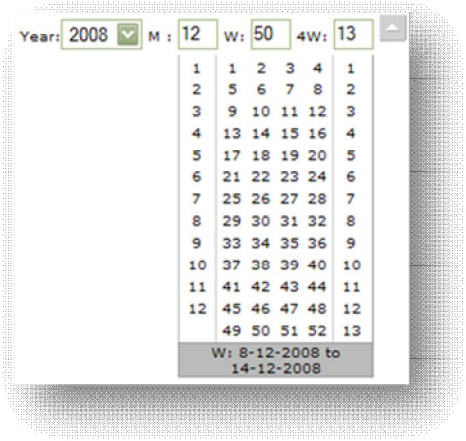


Figure 11: Period picker (expanded)

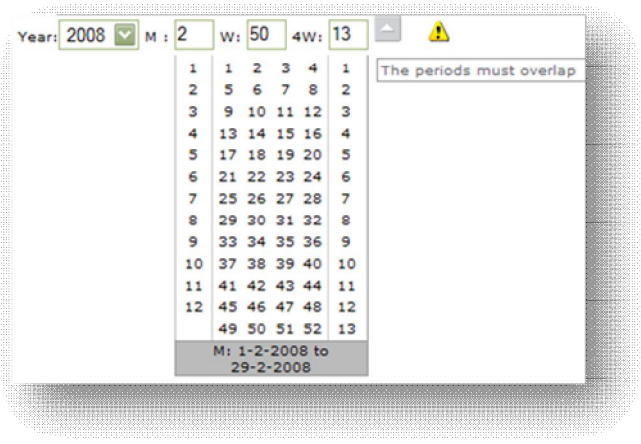


Figure 12: Period picker (warning)

Dropdown extender

Sometimes the user needs to add items to a dropdown list. That is the idea because this dropdown extender component comes up. With it, the user can add new items to a dropdown and delete items that he added before. The user could never delete the default items.

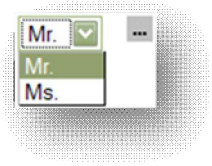


Figure 13: Dropdown extender (collapsed)

Figure 14: Dropdown extender (expanded)

5.4.3 Evaluation

With this prototype it is proved that it is possible to build what was described on the second prototype. And what is more, it works and the client is really happy with it. The application is very intuitive and easy to use.

The most convenient prototype for usability testing is an early working version of the system and this is what this third prototype means. In this prototype the style has also been taken into account, making the application nicer. Different themes can be applied in order the user chose the application color he likes the more and the login page has been built. All this is to make a working system which brings a sense of realism to test tasks.

This third prototype is ready to handle user evaluations as evaluators can rely on the system to guide users through the test tasks. Issues associated with overall complexity or internal consistency can be examined, and the measures of satisfaction or irritation that are collected will be much more meaningful.

Chapter 6. *Technical Design*

6.1 Overview

This chapter is born as the need to explain some technical components and their design. At the time these components were needed, some research on internet was made looking for something already done; but nothing was found which was good enough, so these components have to be developed. The following sections explain these components in a similar way to the ones on the site www.welie.com, in which first, the problem that a user can find is described to later introduce the component as a solution to that problem.

6.2 Dashboard

Problem

Show a big amount of information when some information is relevant for some users but not for others.

Solution

Set blocks of information into different containers with overviews and a bar on the bottom of each container to go into another page which contains that information and more details.
These blocks of information can be removed/added from/to the Web page, drag and drop in a different position on the Web page and collapsed or open according to the user preferences.
There is also a panel in which the names of all the possible containers missing on the Web page are shown, so the user can add them again into the Web page.
Furthermore, each container is divided in different areas that could have a default action triggered when clicking on it, as edit the information.

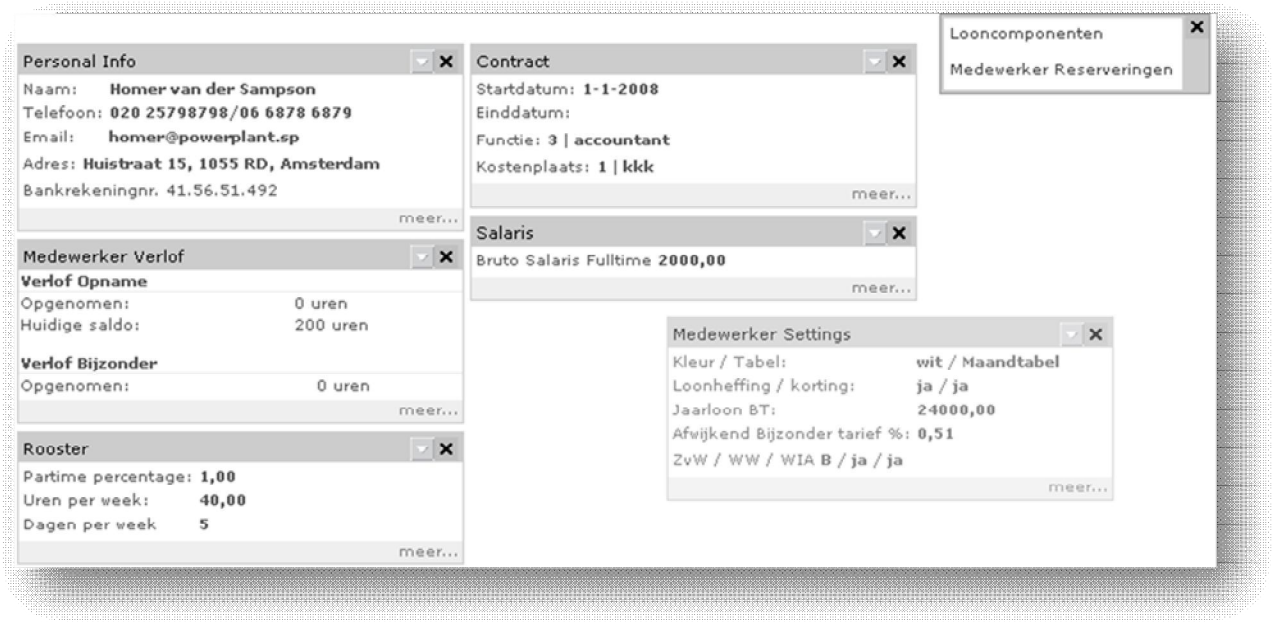


Figure 15: Dashboard component

Used when

The users want to have a start page with a quick information overview and the possibility of going into a details page where more information is shown and the main functionality is placed.

The users want to customize the Web pages directly from the browser:
– Remove or add blocks of information.

– Fit information blocks on a different place on the Web page.
– Collapse information blocks.

Implementation

It consists on the following components:
– Dashboard Control
– Dashlet.js
– Dashlet Manager

Client Side

Dashboard Control: It is placed on each content page which needs a Dashboard. It contains the Dashboard framework which consists of:
– A *div* to open the dashlets catalog.
– A *div* for the dashlets catalog.
– A *table* with one row and three columns (zones).

Dashlet.js:
The following functions can be called no matter the server end point.

LoadDashboard (): it is called on the PageLoad event of the Dashboard Control. It makes an AJAX Request getting a list with the dashlet´s numbers per zone and on the response will call the LoadZones (list) function.

LoadZones (list): this is the function that loads each Dashlet on its correspondent zone calling the function LoadWebPart per Dashlet. Each Dashlet will be asynchrony loaded.

LoadDashLet(ID, Zone): It makes an AJAX Request to get the HTML content of the Dashlet. It uses the function CreateSortable () to make the Dashlet draggable and droppable. This function is part of the Scriptaculous framework (http://script.aculo.us/).

The code of these functions is show on the Appendix D.

Server Side

Dashlet Manager: On the server side this manager is needed to get the HTML for a certain Dashlet and send it back to the client.

The following diagram shows how a Dashboard is loaded

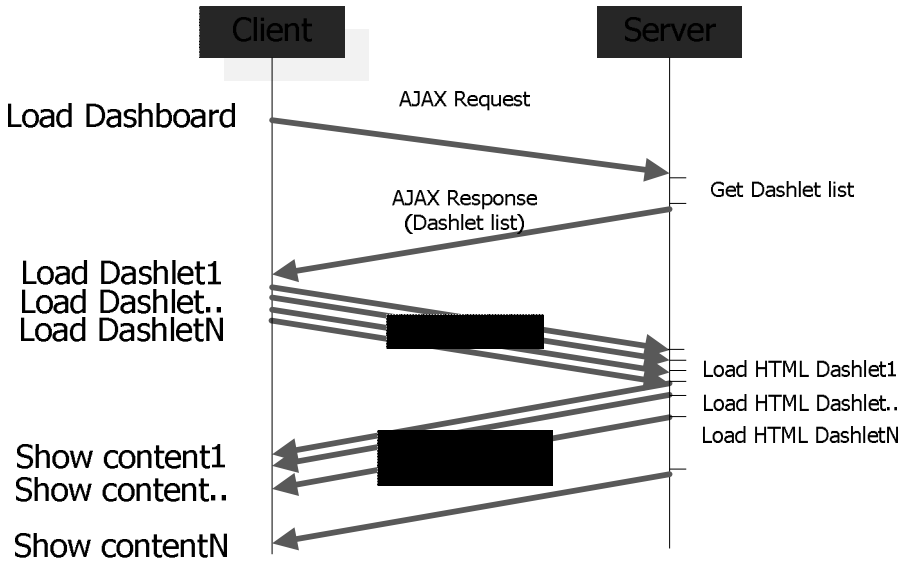


Figure 16: Load Dashboard

6.3 Popup

Problem

Change or add information without forgetting about saving.

Solution

Use a popup window that prevents the user from interacting with the rest of the page and can only be closed by either saving the changes or canceling.

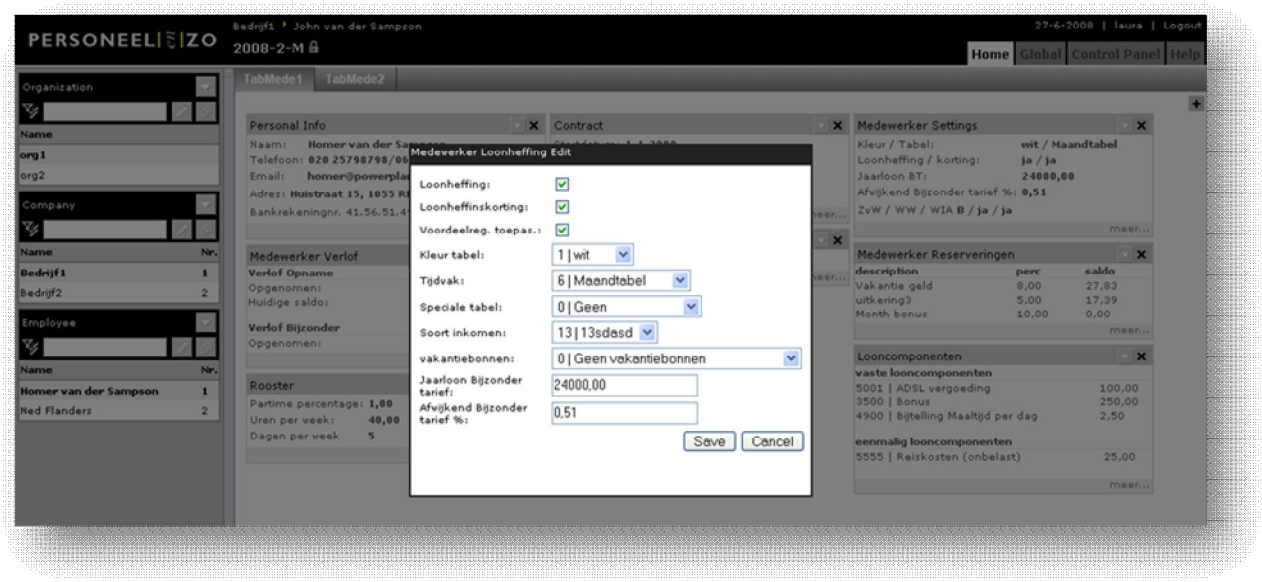


Figure 17: Popup component

Used when

There are too many possible popups to be opened and the page cannot contain all those popups´ code because it would take too much time the page to load.

Implementation

It consists on the following components:

- Popup Control
- Popup.js
- Popup Manager

Client Side

Popup Control: It is placed on the master page and contains the popup framework which consists of:

- A *background div* external to the popup window to avoid the user to interact with anything else but the popup itself.
- A *div* for the popup window.
- A *div* to show the cancel button if there is some kind of error when loading the popup content.

Popup.js:
The following functions can be called no matter the server end point.

OpenPopup: it is called when the user tries to open a popup. It makes visible the *background div* and makes an AJAX request to get the popup content.

ClosePopup: to close the popup window.

Server Side

Popup Manager: On the server side this manager is needed to load the HTML for certain popup content and send it back to the client.

The following diagram shows how a Popup is loaded.

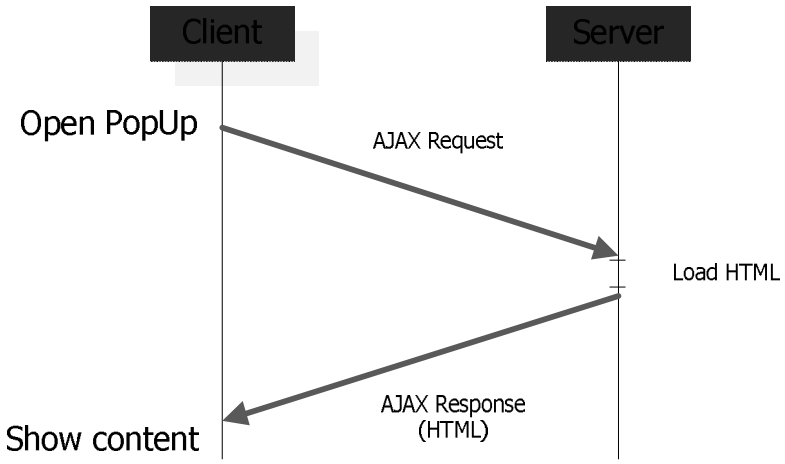


Figure 18: Load popup

Chapter 7. *Conclusions*

In the course of this thesis the interface layer of the web based salary application PersoneelOnline has been developed.

Other applications and some sketches made on Excel files have been the base up on which the interface has been built.

The aim of this thesis was not only to develop the interface but also to study its usability in order to make it as user friendly as possible. It had to be a quick and easy application; quick because the users should accomplish their tasks as fast as possible and without errors and easy because the user should feel comfortable working on it. Although PersoneelOnline is a very complex salary and personnel management system the main goal of this thesis is to make the user feel as if everything was under his control; he should never feel lost working with this tool.

To make all these possible the different scenarios in which this application is going to be used have been analyzed. It is very important to know all the possible situations on which this application can be run as it is very important to build an interface in which every user, on his own scenario, feels like this tool is made for him.
Some interviews have been developed and taken to the different stakeholders. This way have been analyzed the advantages, disadvantages and usability issues of the tools in use as well as the expectations for the new system under developed.

With the interviews outcomes and the interface Excel file sketches the prototyping phase of the project started. This is the most important part of this thesis as is on it where the prototypes are developed and where the evaluations to these prototypes are done. The prototypes have been built in an iterative and progressive way.
The first one was based on the Excel file sketches and although the final result has nothing to do with this first prototype it has been very useful to get in contact with the new application and to know what was it about.
It is on the second prototype where the application starts taking form. In fact, the second prototype is a horizontal prototype in which all the application features are shown but without going into technical details. This second prototype started when the evaluation of the first one finished. This evaluation took quite a long time as the first prototype was very far away from the idea for the new application. With the problems and expectations founded on the interviews and after several brainstorming sessions together with the client and developers the evaluation of the first prototype finished and the second prototype started.
However, it is not that easy to set a line in which the second prototype evaluation finished and the third prototype started. The third prototype is what it´s called a *proof of concept* or *vertical* prototype. On it, all the features defined on the second prototype have been implemented and is during this implementation where the problems came up and solutions had to be found to solve them.

The result of six month of work is the third prototype of the new PersoneelOnline. In this prototype it can already be seen the future system and the goal of the thesis has been reached. The interface is what it was expected to be and it is ready to handle possible system extensions as everything on it is very well defined.

Appendix A. Task Analysis

Appendix A.1 *Standard Roles, User Levels and Tasks*

Standard Roles

(R1) Pay roll Specialist at Personeel en Zo
(R2) Pay roll Specialist of Clients
(R3) Normal User
(R4) Pay roll Specialist of one client
(R5) HR Specialist
(R6) Employee
(R7) Debtor

User Levels

(L1) Administrator
(L2) Client
(L3) Group
(L4) Organization
(L5) Company
(L6) Employee
(L7) Debtor

Tasks per Role and User Level

Group settings

- Insert Group (example accountant office)
  - Bind organizations to a Group
- Insert Organization (example big company 100+)
  - Bind companies to an organization
  - Insert Departments (example system administration)
- Insert Debtor
  - Bind a Group to a debtor
  - Bind a organization to a debtor
  - Bind companies to a debtor
- Insert a System administrator (professional Personeel en Zo) … *L1*
  - Insert username
  - Insert password
  - Insert email address
- Insert a new group user (accountant) …… *L2*
  - Insert username
  - Insert password
  - Insert email address
  - Bind to a Group
  - Bind to an authorization profile
- Insert a new organization user …… *L3*
  - Insert username
  - Insert password
  - Insert email address
  - Bind to an organization
- Insert a new company user …… *L4*
  - Insert username
  - Insert password

  - Insert email address
  - Bind to a company
- Insert a new department user …… *L5*
  - Insert username
  - Insert password
  - Insert email address
  - Bind to a company
- Insert a new employee user ……. *L6*
  - Insert username
  - Insert password
  - Insert email address
  - Bind to a organization / company / department
- Insert Debtor user …….…… *L7*
  - Insert username
  - Insert password
  - Insert email address
  - Bind to a debtor
  - Bind to an authorization profile
- Insert authorization profiles (none/read / write)
  - Rights for the pages

Login page

- Login
- Recover password by email

Start page

- Modify user information (login name / password / email address)
- Module Agenda (signals, birthdays, important dates)
- Module illness (pending illness cases)
- Module absence (people on holidays)
- Module wage costs (brief survey salary cost)
- Quick link to a module (list with activities)

System settings

- Insert a loon codes …….. L1 R1
- Insert a pension plan …….. L1 R1, R2
- Insert model company …… L1 R1
- Insert a salary table …… L1 R1
- Insert a SVW table (franchise / percentages. L1 R1
- Insert a SVW table (sector and risk group)…. L1 R1

Debtors

- Insert a new debtor …… L1 R1, R2
- Insert a debtor contract …….. L1 R1
- Manage debtor NAW information …… L7 R1, R2, R3, R4, R5, R7

Organization management

- Insert an organization ……..…… L2 R1, R2, R3, R4, R5
- Insert an organization contact person, email ……. L3 R1, R2, R3, R4, R5

- Insert departments ……. L3 R1, R2, R3, R4, R5
- Insert departments contact person / email ……. L3 R1, R2, R3, R4, R5
- Insert external suppliers (bijv. arbodienst) / email …… L3 R1, R2, R3, R4, R5
- Insert a contact information for the external suppliers (*)/ email L3 R1, R2, R3, R4, R5
- Insert the company free days (calendar) …… L3 R1, R2, R3, R4, R5

Company management (salary registration)

- Insert a new company ……... L3 R1, R2, R3
- Insert period type (week/4weeks/months) …… L4 R1, R2, R3, R4
- Insert a company contact person / email …….. L4 R1, R2, R3, R4, R5
- Insert a cost location …… L4 R1, R2, R3, R4
- Modify the company SVW table (bvWGA premie) …… L4 R1, R2, R3, R4
- Insert a Loonhefing nummer …… L4 R1, R2, R3, R4
- Import new employees ……. L4 R1, R2, R3, R4, R5
- Insert a company Loonmodel …… L4 R1, R2, R3, R4
- Bind the looncodes to a company grondslagen …… L4 R1, R2, R3, R4
- Insert model employee(s) ……. L4 R1, R2, R3, R4
- Insert a accountant book / descriptions ……. L4 R1, R2, R3, R4
- Insert a reservation …… L4 R1, R2, R3, R4
- Insert components to see on the standard register overview …… L4 R1, R2, R3, R4
- Bind a company to a pension plan …… L4 R1, R2, R3, R4
- Insert the company pension plan ……... L4 R1, R2, R3, R4
- Insert a salary table to a company ……. L4 R1, R2, R3, R4, R5
- Make available a salary table to an employee …….. L4 R1, R2, R3, R4, R5
- Insert employee functions to a company …….. L4 R1, R2, R3, R4
- Bind the looncodes to the accountant book …….. L4 R1, R2, R3, R4
- Copy a company ……... L2 R1, R2, R3, R4
- Copy a company with employees ……. L2 R1, R2, R3, R4
- Copy a company with employees and period type …….. L2 R1, R2, R3, R4
- Quick insert looncomponents …….. L4 R1, R2, R3, R4
- Import looncomponents …… L4 R1, R2, R3, R4

Employee salary management

- Insert a new employee …….. L5 R1, R2, R3, R4, R5
- Insert Eerste Dag Melding (EDM) …… L5 R1, R2, R3, R4, R5
- Out of service notification for an employee ……. L5 R1, R2, R3, R4, R5
- Modify Personal information …… L5 R1, R2, R3, R4, R5
- Modify Bank account …… L5 R1, R2, R3, R4, R5
- Modify the calendar …….. L5 R1, R2, R3, R4, R5
- Modify the Salary …… L5 R1, R2, R3, R4, R5
- Modify the Contract ……. L5 R1, R2, R3, R4, R5
- Modify the Leasecontract ……. L5 R1, R2, R3, R4, R5
- Modify Pension plan (participation yes/no) …… L5 R1, R2, R3, R4, R5
- Modify a Reservation (participation yes/no) …… L5 R1, R2, R3, R4, R5
- Modify loon components …… L5 R1, R2, R3, R4, R5
- Modify loonhe_ng. …… L5 R1, R2, R3, R4, R5
- Modify components to see on the standard register overview …. L5 R1, R2, R3, R4, R5
- Modify the SVW (deelname ja/nee) . ……... L5 R1, R2, R3, R4, R5
- Modify insurance policy ……... L5 R1, R2, R3, R4, R5
- View the current payslip ……... L5 R1, R2, R3, R4, R5
- Make a Proforma calculation. ……... L5 R1, R2, R3, R4, R5

Employee staff management

• Modify department / position ……	L5	R1, R2, R3, R4, R5
• Modify holidays ……	L5	R1, R2, R3, R4, R5
• Set the amount of holidays ……	L5	R1, R2, R3, R4, R5
• Modify absence ……	L5	R1, R2, R3, R4, R5
• Insert illness report ……	L5	R1, R2, R3, R4, R5
• Modify Dossier (action points) ……	L5	R1, R2, R3, R4, R5
• Modify Dossier (documents) ……	L5	R1, R2, R3, R4, R5
• Insert user information for employee login ……..……	L5	R1, R2, R3, R4, R5
• Send user information for employee login ……	L5	R1, R2, R3, R4, R5
• Modify supplies / loan information ……...……	L5	R1, R2, R3, R4, R5
• Merge MS word files (contract / bruikleenovk. Etc.) …..……	L5	R1, R2, R3, R4, R5
• Module recruitment en selection ……...	L5	R1, R2, R3, R4, R5
• Modify training plans (*) ……	L5	R1, R2, R3, R4, R5
• Modify Competences (*) ……	L5	R1, R2, R3, R4, R5
• Modify Insurances (*) ……	L5	R1, R2, R3, R4, R5
• Modify family / partner and children information. (*) ……	L5	R1, R2, R3, R4, R5
• Modify evaluation (*) ……..	L5	R1, R2, R3, R4, R5
• Modify external contacts (family doctor / alarm number) (*) …..	L5	R1, R2, R3, R4, R5
• Modify phisical exams (*) ……...	L5	R1, R2, R3, R4, R5
• Dashboard staff information (*) ……...	L5	R1, R2, R3, R4, R5

Salary processing

• Process the salary run for a company …….	L4	R1, R2, R3, R4
• Roll back the salary run for a company ……	L4	R1, R2, R3, R4
• Proccess the salary run for a company t.b.v. TWK calculations...	L4	R1, R2, R3, R4

Salary outputs View / generate

• Proccess Status overview/ niet verwerkte loonruns per company ……..	L4	R1, R2, R3, R4
• View / Print pay slip per company ……..	L4	R1, R2, R3, R4
• View a Historic overview per company / per employee …….	L4	R1, R2, R3, R4
– View / Print van payment list per company ……	L4	R1, R2, R3, R4
• View / Print standard register component per company …….	L4	R1, R2, R3, R4
• View / Print wage costs per company/employee/ cost location ……..	L4	R1, R2, R3, R4
• View / Print Year statement ……	L4	R1, R2, R3, R4
• View / Print Journal overview per company/employee/department ……	L4	R1, R2, R3, R4
• View / Print deductions and payments of the pension plans (period type)	L4	R1, R2, R3, R4
• View / Print fiscal overview ……...	L4	R1, R2, R3, R4
• Insert Clieop file for bank transfers ……	L4	R1, R2, R3, R4
• Insert wage declaration XML …….	L4	R1, R2, R3, R4
• Generate a .pdf with all monthly output …….	L4	R1, R2, R3, R4
• Dashboard financial (*) …….	L4	R1, R2, R3, R4

Signals per email:

• Manage of signals (who receives which signals).....……...	L5	R1, R2, R3, R4, R5
• Birthday ……...	L5	R1, R2, R3, R4, R5
• End contract …….	L5	R1, R2, R3, R4, R5
• End probation ……..	L5	R1, R2, R3, R4, R5
• Expiration day for the passport ……...	L5	R1, R2, R3, R4, R5
• Action point of staff file ……..	L5	R1, R2, R3, R4, R5
• Action point of absence ……...	L5	R1, R2, R3, R4, R5

Staff information overview / generate

- View / Print vacation overview ……... L5 R1, R2, R3, R4, R5
- View / Print absence overview …… L5 R1, R2, R3, R4, R5
- View / Print N.A.W. overview …… L5 R1, R2, R3, R4, R5
- View / Print birthday calendar (*)…… L5 R1, R2, R3, R4, R5
- View / Print Organigram (*)…… L5 R1, R2, R3, R4, R5

Module wage declaration

- View / Send wage declaration XML …… L4 R1, R2, R3, R4
- View / Send wage declaration EDM …… L4 R1, R2, R3, R4

Overviews

- Overview imported mutations per user ……... L4 R1, R2, R3, R4, R5
- Overview imported mutations per company …….. L4 R1, R2, R3, R4, R5
- Status overview (not) processed loonruns ……. L4 R1, R2, R3, R4, R5
- Status overview of sent signals …….. L5 R1, R2, R3, R4, R5

Employee logins

- Insert company settings of the employee login ……... L4 R1, R2, R3, R4, R5
- Insert (employee) user ...…….. L4 R1, R2, R3, R4, R5
- Insert user profiles ……... L4 R1, R2, R3, R4, R5
- View employee holidays overview ……. L6 R1, R2, R3, R4, R5
- View / Modify employee personal information …….. L6 R1, R2, R3, R4, R5
- View / Modify payslip individual employee ……... L6 R1, R2, R3, R4, R5
- Leave card claim ……. L6 R1, R2, R3, R4, R5
- Holiday planning ……. L6 R1, R2, R3, R4, R5
- Hours registration justification …… L6 R1, R2, R3, R4, R5
- Insert Declarations ……... L6 R1, R2, R3, R4, R5

Support pages

- Getting started
  - Insert company
  - Insert employee
  - Insert looncomponent
- Quick help per tab.
  - FAQ

Appendix A.2 *Interviews.*

In this section the interviews taken to the different stakeholders are shown. The first one, Internal Users Interviewing Guide, was used to all the stakeholders but the Pay roll specialist of clients. To this particular stakeholder the Manager Interviewing Guide was used. A particular case of this stakeholder group is the manager of the company which is developing this application and another kind of questions as the project goals are more interesting on this case.

Internal Users Interviewing Guide

Here you have some questions I need to know about you as future user of the PersoneelOnline system. Please, answer as much as you can and take the time you need. I prefer good rather than fast answers.

1. Open-ended questions.
    a. How would you describe yourself as a user? (User characteristics, i.e., age, experience, education, etc.)
    b. What background do you have in using computers?
2. Talking About Cobra system
    a. How long have you been working with this application?
    b. Why do you use this application? (Needs, interests, and goals)
    c. When and where do you access the system?
    d. How do you access the system? (Connection speed, resolution, etc.)
3. Talking About PZ Web system
    a. How long have you been working with this application?
    b. Why do you use this application? (Needs, interests, and goals)
    c. When and where do you access the site?
    d. How do you access the site? (Connection speed, resolution, etc.)
4. Conduct task analysis and prioritize tasks
    a. What do you do on the system? (Tasks, content, features and functionality)
    b. Which tasks are critical to the organizations success on the system?(Criticality)
    c. Which tasks are critical to your success on the system?
    d. Which tasks are most important to you? (Importance)
    e. Which features of the system do you use the most? (Frequency)
    f. Which features are prone to usability issues? (Vulnerability)
5. Measurable usability objectives
    a. Which tasks should you be able to accomplish easily with few errors?(Efficiency)

b. Which tasks should you be able to finish quickly and efficiently?(Effectiveness)

c. What level of satisfaction should you have after using the system?(Enjoyability)

d. What level of satisfaction do you actually have?

6. Expectations, requirements and preferences regarding the new PersoneelOnline

   a. What is your vision of what the system should do?

   b. Would you add, change or remove something in the new system? (Tasks, content, features or functionality)

   c. Are there any interface styles that you prefer or feel very comfortable with? Give some real examples.

7. Any other comments

Thank you for your time!

Manager Interviewing Guide

Here you have some questions I need to know about you as future user of the PersoneelOnline system. Please, answer as much as you can and take the time you need. I prefer good rather than fast answers.

1. Open-ended questions
    a. How would you describe yourself as a user? (User characteristics, i.e., age, experience, education, etc.)
    b. What background do you have in using computers?
2. Talking About PZ Web system
    a. How long have you been working with this application?
    b. Why do you use this application? (Needs, interests, and goals)
    c. When and where do you access the system? How do you access the system? (Connection speed, resolution, etc.)
3. Develop goals for the system
    a. How would you define a successful system for your organization?
    b. What does success look like?
    c. How will you know when you have been successful?
    d. How would you describe the system
        i. From an organizations viewpoint?
        ii. From a users viewpoint?
4. Conduct task analysis and prioritize task
    a. What do you do on the system? (Tasks, content, features and functionality)
    b. Which tasks are critical to the organizations success on the system?(Criticality)
    c. Which tasks are critical to your success on the system?
    d. Which tasks are most important to you? (Importance)
    e. Which features of the system do you use the most? (Frequency)
    f. Which features are prone to usability issues? (Vulnerability)
5. Measurable usability objectives
    a. Which tasks should you be able to accomplish easily with few errors?(Efficiency)
    b. Which tasks should you be able to finish quickly and efficiently?(Effectiveness)
    c. What level of satisfaction should you have after using the system?(Enjoyability)
    d. What level of satisfaction do you actually have?
6. Purpose/vision of the new PersoneelOnline

a. What is the purpose of the system?

b. What are the goals of the system?

c. What prompted the redesign?

7. Expectations, requirements and preferences regarding the new system

a. What is your vision of what the system should do?

b. Would you add, change or remove something to the new system?(Tasks, content, features or functionality)

c. Are there any interface styles that you prefer or feel very comfortable with?

d. Give some real examples.

8. Any other comments

Thank you for your time!

Appendix A.3 *Interviews outcomes*

In this section are shown the interview outcomes per stakeholder.

Payroll Specialist at Personeel en Zo (Internal Users Interviewing Guide)

1. *Open-ended questions*
   - Payroll administrator.
   - From 25 to 45 years old.
   - Several years as payroll administrator.
   - Working with different systems as ADP, CMG, SAP CRM and SAP HR.
   - Currently working with Cobra.
   - ADP is more trustable than Cobra is.
   - Cobra is less trustable but you can work on it at anytime.
   - They use Excel for making calculations.
   - They are used to work with computers at user level.

2. *Talking about Cobra system*
   - Working with Cobra since 2006.
   - They use Cobra to make all the pay slip's calculations, set everything about a new company, establish the social security settings, make corrections, proformas ...
   - They access the system at work.
   - The server is not in their office. They have to access the system through a remote desktop which makes this access slowest.
   - They consider Cobra fast and friendly but they would like it to be more friendly and trustable.

3. *Talking about PZ Web system*
   - They know how PZ Web works and although they only work with Cobra, some of them started working with PZ Web.
   - PZ Web is the portal. The clients use it to make requests and Personeel en Zo have to process and answer these request. They use PZ Web to support the client walking with him throw the system.
   - Personeel en Zo consider easier to see the XML corrections in PZ Web than in Cobra.
   - They access the system mainly at work but also at home for doing some things as checking the email.
   - They would like to have a Data Management System (DMS) in which they wouldn't have to make calculations anymore. The clients would make the requests and the system, let's say would automatically generate the outcome of that request (Not manual anymore).

4. *Conduct task analysis and prioritize tasks*
   - They use the system to
     - Implement new clients.
     - Processing the salary.
     - Make changes in the setting where the client is not allowed to access.
     - Make analysis.
     - Download reports in Excel and make analysis (statistics).
     - Make offers for a company when it is interested on the system.
     - Put back the payroll period. This is something that goes throw a manual process and can lead to errors. When they have to put back a payroll period the communication between the payroll administrators is throw email or telephone which is no secure enough.
   - They consider critical tasks
     - The deadline of the payroll administration (critical dates).
     - What happened if the company grows? Maybe they would need more payroll administrator during the critical days.
     - Need of DMS.

  - Speed of the system. Problems with the system most of the time are due to updates or the connection with the external server.
  - The most important for them as users is that the system was fast and trustable. (Trustable pay slips on time)
- The most important task for them is to process pay slips.
- And the features on the system they use the most are the "batch" and the loon components.
- About the features which are prone to usability issues..
  - The Save button.
  - Don't repeat the same information several times.
  - Group tasks in different pages.
  - Like salary experts they like to see several tabs with all the functionalities on one page, but like users, it scares them.

5. *Measurable usability objectives*

6. *Expectations, requirements and preferences regarding the new PersoneelOnline*
   - They found that the system should be easy to use without too much problems. The whole process shouldn't take too long (the monthly payroll, the payroll taxes, etc.)
   - They consider the system quite slow sometimes. This is the biggest problem. If it works at normal speed, the system works fine.
   - Another problem is the updates: after the updates, there are some errors in the system; for example, some documents can't be printed.
   - If there is a mutation in the salaries of CAO (see CAO code), it would be good if you only need to change this in one place and the other companies would change it automatically.
   - They would like to have a list in which you can see when a company is already implemented.
   - To use DMS. Year Management Documentation is also manual.

Payroll Specialist of clients (Manager Interviewing Guide)

1. *Open-ended questions*
   - From 35 to 45 years old.
   - Normal office applications.
   - Payroll specialist.

2. *Talking about PZ web system*
   - 3 to 4 years working with PZ Web.
   - They use this application to provide the payroll administration to their customers.
   - They access the site from their own office or when visiting clients.

3. *Develop goals for the system*
   - Easy to use.
   - Access for every location.
   - Give their customers also access to the system.
   - They want to see what a customer has done on the system.
   - Speed in processing. Now they have to wait until a payroll specialist process their request and send them the outcome.
   - Be able to process everything at every time.
   - They describe the system like a Payroll processing system.

4. *Conduct task analysis and prioritize tasks*
   - With the new system they want:
     - To see what a customer has done on the system.
     - Speed in processing. Now they have to wait until a payroll specialist process their request and send them the outcome.
     - Be able to create/modify companies/employees.
     - Be able to make proformas.
     - Process the pay slips themselves.
     - They would like to be able to have in only one file all the documents to print.
     - Manage signals.
     - Give their clients access to the system.
     - Enter/Modify companies/employees.
     - Make proformas.
   - Critical tasks are:
     - Immediately payroll process.
     - Make fewer mistakes.
   - The most important for them is the payroll processing system.
   - And they also find very important to be able to make proformas.
   - The feature on the system they use the most is managing loon components (Add/Modify)

5. *Measurable usability objectives*
   - They should be able to add new employees and process the "out of service" tasks easily with few errors.
   - The task they should be able to finish quickly and efficiently is entering loon codes.
   - To increase their level of satisfaction they would like to know the result of what they are doing immediately.

6. *Purpose/vision of the new PersoneelOnline*
   - They see the new system like a very efficient payroll administration system in which they can make everything on their own.
   - The goal of the system will be to allow the users processing pay slips themselves

7. *Expectations, requirements and preferences regarding the new PersoneelOnline*
   - Login of the mutations.
   - Give their clients the feeling that it is Their System

Normal User (Internal Users Interviewing Guide)

1. *Open-ended questions*
   - From 22 years old.
   - Grew up with computers and use it every day.
   - Have all different kinds of other high Tec products like the iPod and mobile phone.
   - Use the computer to get all kinds of information and to communicate.
   - They learned to use computer programs like excel, word, internet, on their education.

2. *Talking About Cobra system*
   - They don´t work with this system.

3. *Talking about PZ web system*
   - It took them like a month to really know this system.
   - They mainly use this application to provide their employees the pay slips.
   - It is very good to know that everything works out to the tax company.
   - It is very handy to have access to the system everywhere and always can see all the need information like (verlof) when the employees are on holidays or for how long they were ill.
   - It is also a good application because you can put all kind of information in the system: employees ID, contract, results of a meeting...
   - They access the system mainly at work but also at home for doing some things as checking the email or have a quick view to employees' information as possible mutations.

4. *Conduct task analysis and prioritize tasks*
   - One of the things they do on the system is: when they are on the system they have a list where all the different mutations of the past month are listed. They copy these mutations in the PZ Web. When they finish this list they ask Personeel en Zo for providing them the pay slips. That is the main task they have.
   - Beside this main task they also use the system to have some insight to all the different employees' information like, spare holidays, illness, etc. They use it also as a way of communication with their employees.
   - They consider critical tasks
     - That all the salary information was correct. It is very important for the employees to get good information about their salary and that there was a lot of trust in the company. They need confidence in the company.
     - To pay the employees on time.
     - To give to the employees a correct pay slip.
     - For company success it is also very important that everything was correct to the tax company.
     - It is necessary that everything just works out. It is very important that the system provides them to focus on the core business of the company. If they need to spend lots of hours in the system with all kinds of bugs coming out, then they don't need the system. It is very important for the success of the company that all the given functionalities work well.
     - Speed is very important for their personal success. They need to work in a quick and handy way.
     - Usability is also very important. All the information needs to be well organized and if they want to find a specific piece of information they want to find it fast.
   - The most important thing for them is to trust in the system. If there is a problem with the system they want it to be solved very quickly. They don't want to spend a lot of time in the system. They want it to work.
   - The features on the system they use the most are
     - The quick possibility to insert the monthly mutations.
     - To ask for a salary calculation.
     - To see the monthly results.

5. *Measurable usability objectives*
   - The tasks they should be able to accomplish easily with few errors are the not so important tasks like, hanging a result of a meeting or a contract; the vacation questionnaires, etc.

- Personeel en Zo should do the important things like the salary related errors.
- About the salary related tasks, they don't want to spend a lot time on that subject.
- They would like to have good feeling about they did all the salary in a short time.
- They want to have a good feeling having the chance of asking the system for all kinds of support at any time. When they need some information they want to find it really fast.
- All the information should be good organized.
- They feel happy with the system but think there is too much irrelevant information on the main pages of PersoneelOnline. They don't use most of this information.

6. *Expectations, requirements and preferences regarding the new PersoneelOnline*
    - They are satisfied with the PersoneelOnline like it is now. They trust in the system and it works out fine.
    - They would hide all the irrelevant information better. They would like to have fewer options in PersoneelOnline.
    - They would like to get the tabs ordered chronologically. As they use it.
    - They think that the "snelkoppelingen" can be downside to a less important tab.
    - They would like to give to each of their employees his own digital pay slip.
    - They found the interface very direct. All blocks are very straight. They would like it to be a bit more childish.

Payroll Specialist of one client (Internal Users Interviewing Guide)

1. *Open-ended questions*
    - From 40 to 50 years old.
    - Very long time experience.
    - Book keepers.
    - Office applications.

2. *Talking about Cobra system*
    - They don´t work with this system.

3. *Talking about PZ web system*
    - The worked with Cobra at the beginning but they are not using it anymore.
    - They use PZ Web for processing pay slips and always from his office.

4. *Conduct task analysis and prioritize tasks*
    - Enter payroll processing mutations.
    - Fill in loon components.
    - Add and dismiss employees.
    - They used to make Proformas when they were working with Cobra, but not anymore.
    - They find a critical task to compare quickly the differences between the last month and the current one (Standard Register).
    - Very important are, the task of entering loon components and making fast calculations.

5. *Measurable usability objectives*

6. *Expectations, requirements and preferences regarding the new PersoneelOnline*
    - They would like
        - To have the import function
        - To be able to make Proformas.
        - To have an overview about all the changes made by someone else.

Human Resource Specialist of clients (Internal Users Interviewing Guide)

1. *Open-ended questions*
    - From 21 years old.
    - They work as users and administrators.

2. *Talking about Cobra system*
    - They don´t work with this system.

3. *Talking about PZ web system*
    - They find it very easy to work with and friendly.
    - The things they really miss on the system are Evaluations (Statistics).
    - They would like to have the help telephone number somewhere visible in the main page.
    - They want to have a template that includes all the documents they can have about a person. So they have visible all the documents they already have and which ones they could still add if they wanted to. "Suggestions"
    - Previous run so they can check if everything is going ok.
    - Make the panels more clear. For example, when they add a bonus, they don't know if it is going to be positive or negative; so explanations about what are they actually doing should be given to them.
    - Check lists when. for example, creating a new employee

4. *Conduct task analysis and prioritize tasks*
    - The tasks they do on the system are tasks as:
        - Make contracts.
        - Make the EDM.
        - Salary changes.
        - Creating an employee.
        - The Verlof and Verzuim tabs.

5. *Measurable usability objectives*
    - Missing the employee photo.
    - Save several things at the same time.
    - Save button should be always visible to not forget to save changes.
    - Chance of combine several things, as salary, CV, address... in only one sheet in order to print it. Check list to select the documents they want to print at the same time could be added.
    - Have the possibility of checking what you have changed or done.
    - They don't use the tabs since the direct links are in the system.
    - They would like to be able to change the salary in advance, as well as other features.
    - Chance of have a button to "send a letter" in *Personal information*. And have a template letter in which the employee address is automatically copied.
    - If an employee is not working anymore for a company...
        - Indicate that the employee is not working anymore so the system can do all the settings (They are always the same).
        - The system could generate a signal if there are holidays left. What do you want to do with them?

6. *Expectations, requirements and preferences regarding the new PersoneelOnline*

Appendix B.

Appendix B. Prototype description

Appendix B.1 *First Prototype*

Navigation

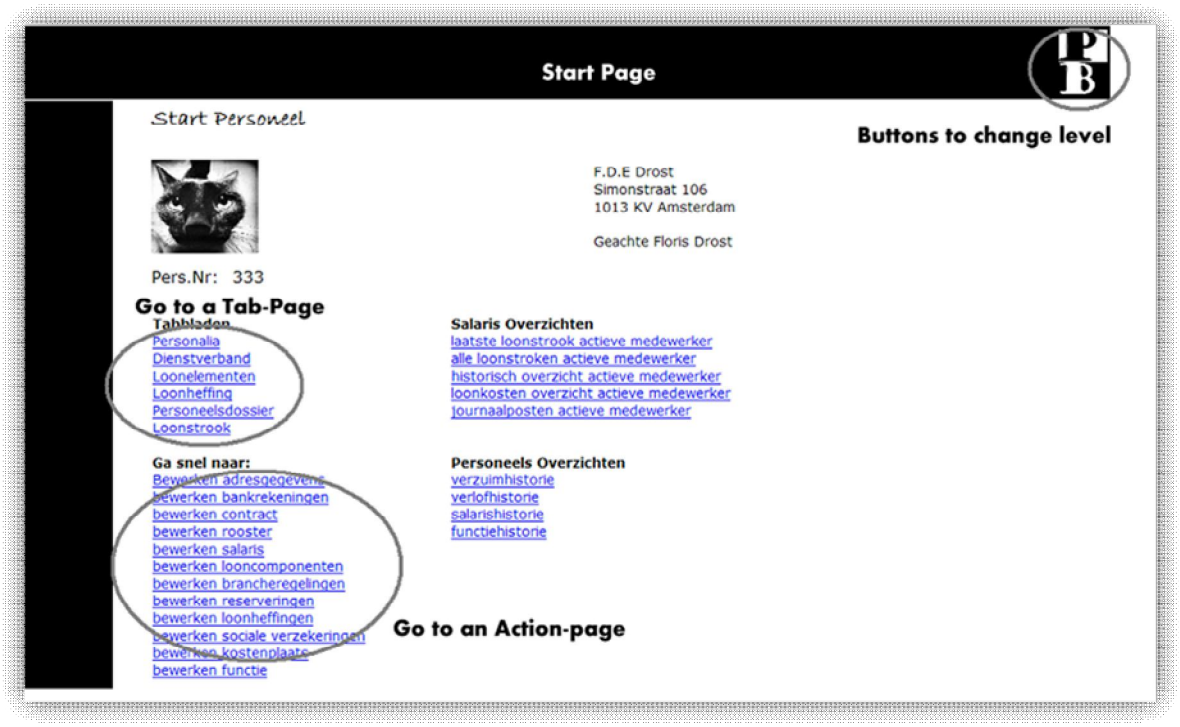


Figure 19: First prototype navigation (Start Page)

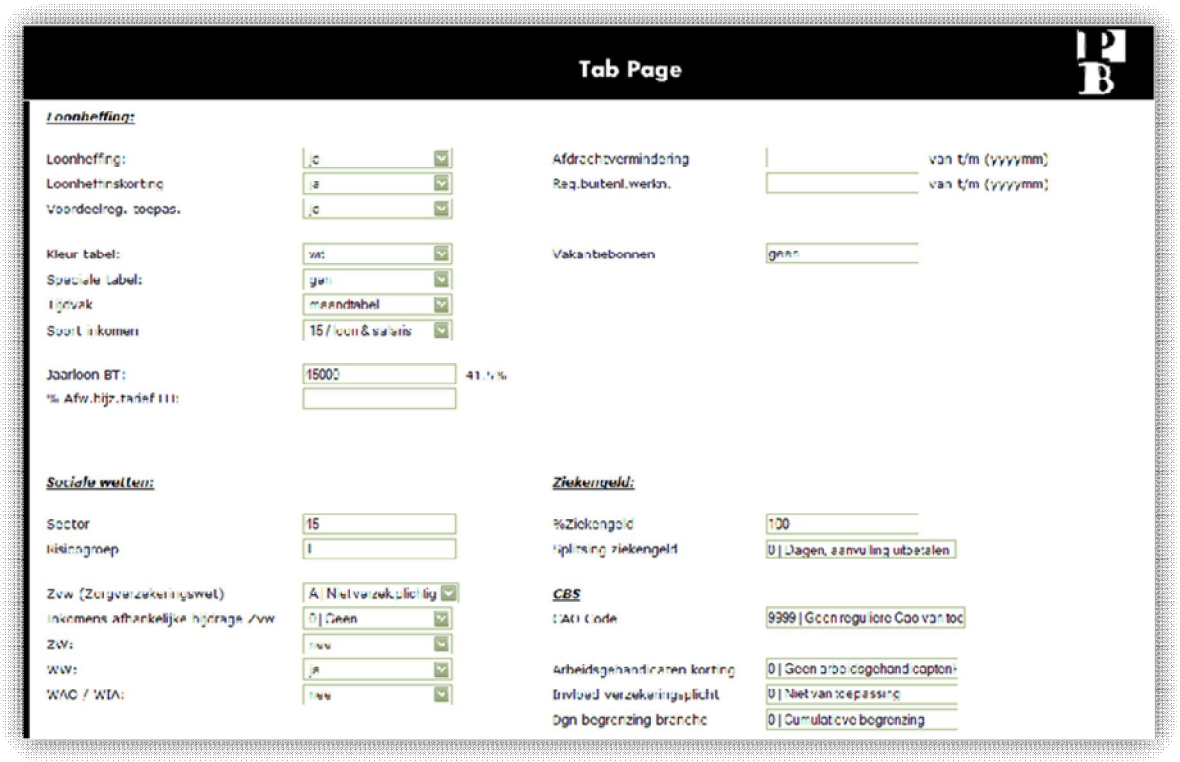


Figure 20: First prototype navigation (Tab-Page)

Figure 21: First prototype navigation (Action-Page)

Evaluation

- Start page with quick links and the company/employee most important information and photo.
- On the left, the left menu to choose between companies and employees will be place on future prototypes.
- The two buttons on the right top corner (*P*,*B*) stand for Personeel and Bedrijf (Employees and Company) and are used to switch between Employee and Company level.

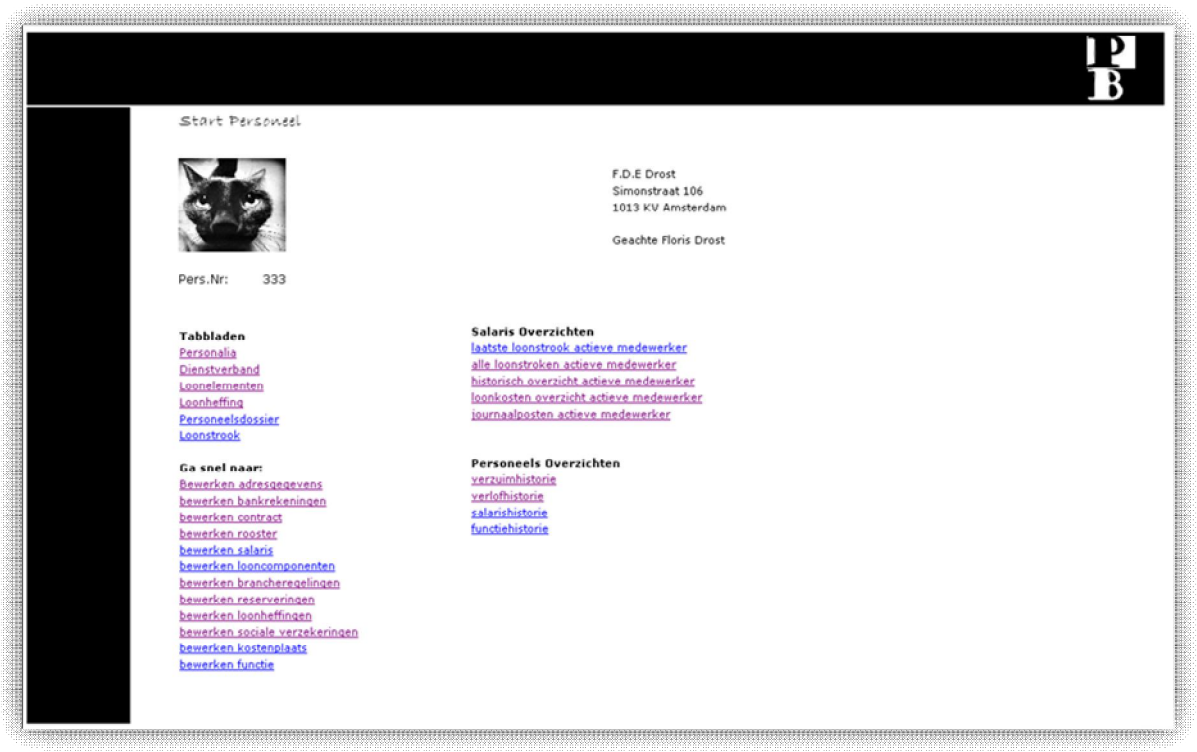
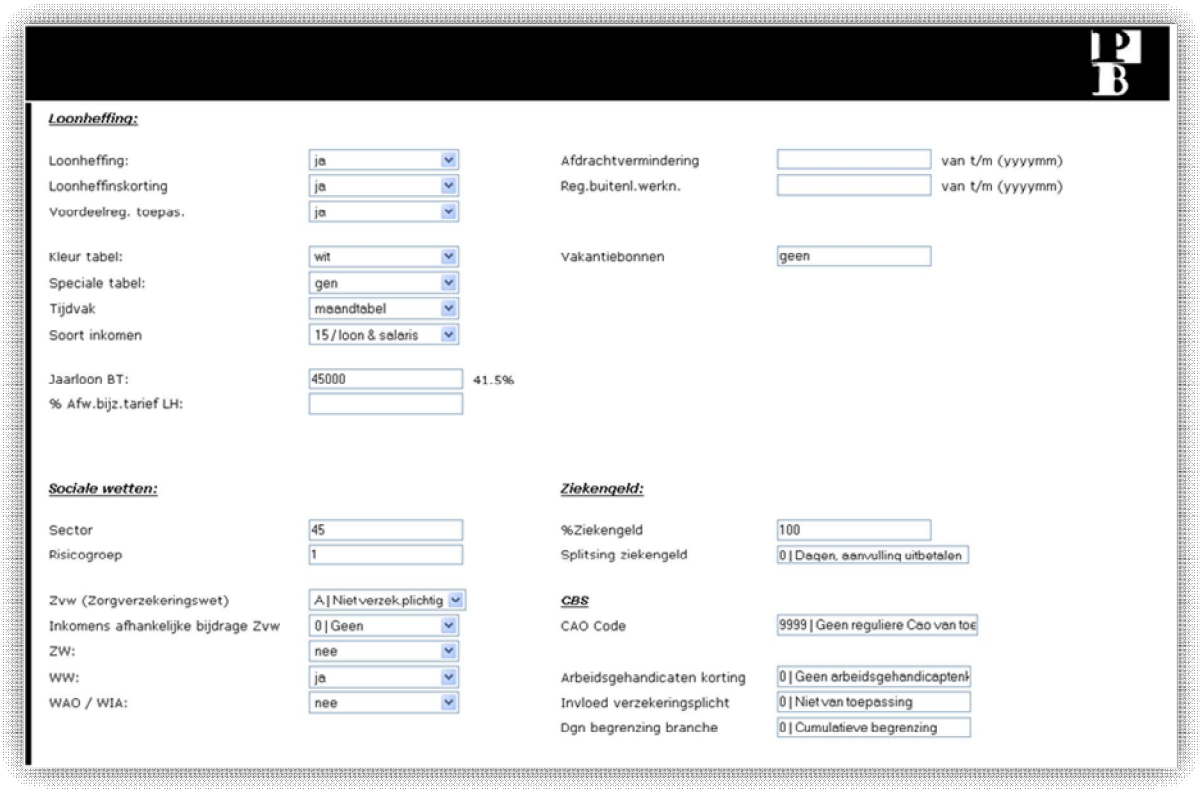


Figure 22: First prototype evaluation (Start-Page)

- These kind of pages are shown either the users want to change the current information or if they only want to see the information.
- The tables and forms are editable and already filled in with the current or default information.



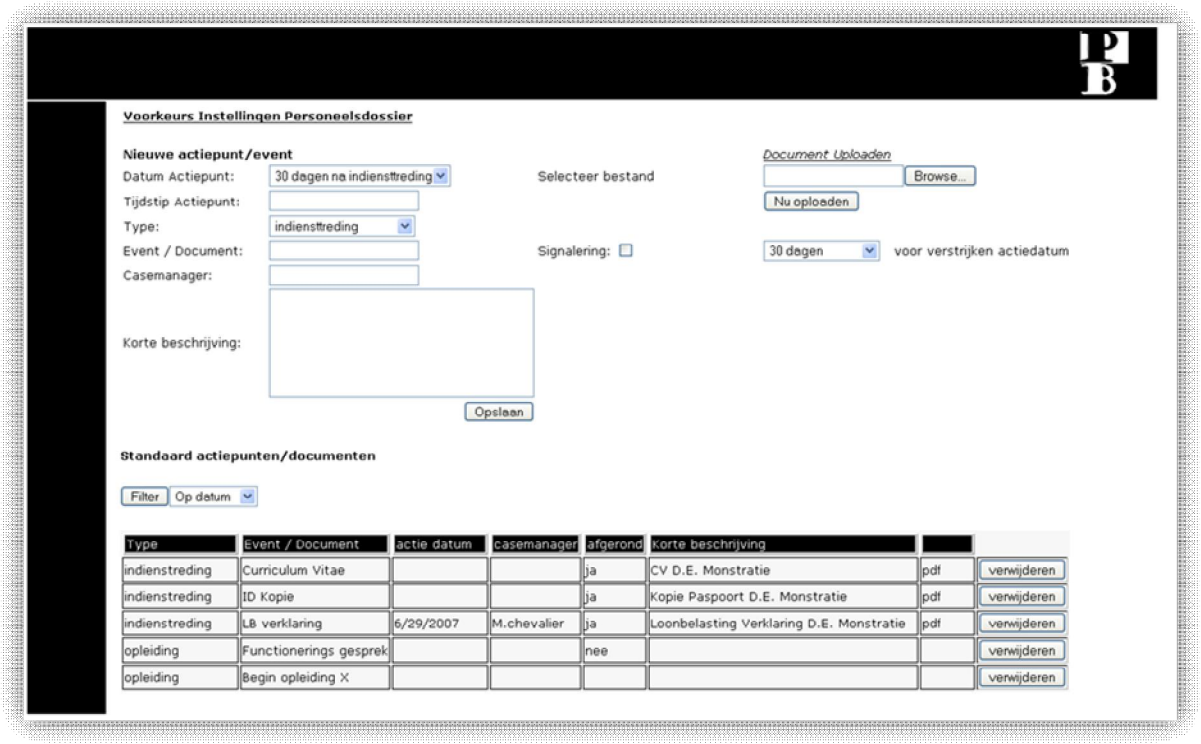


Figure 23: First prototype evaluation (Editable forms)

– Sometimes these pages occupy too much space and the users have to scroll down to see all the information. Due to this if they only want to change something that they can see without having to scroll down, save button is not visible as it is always at the end of the page.

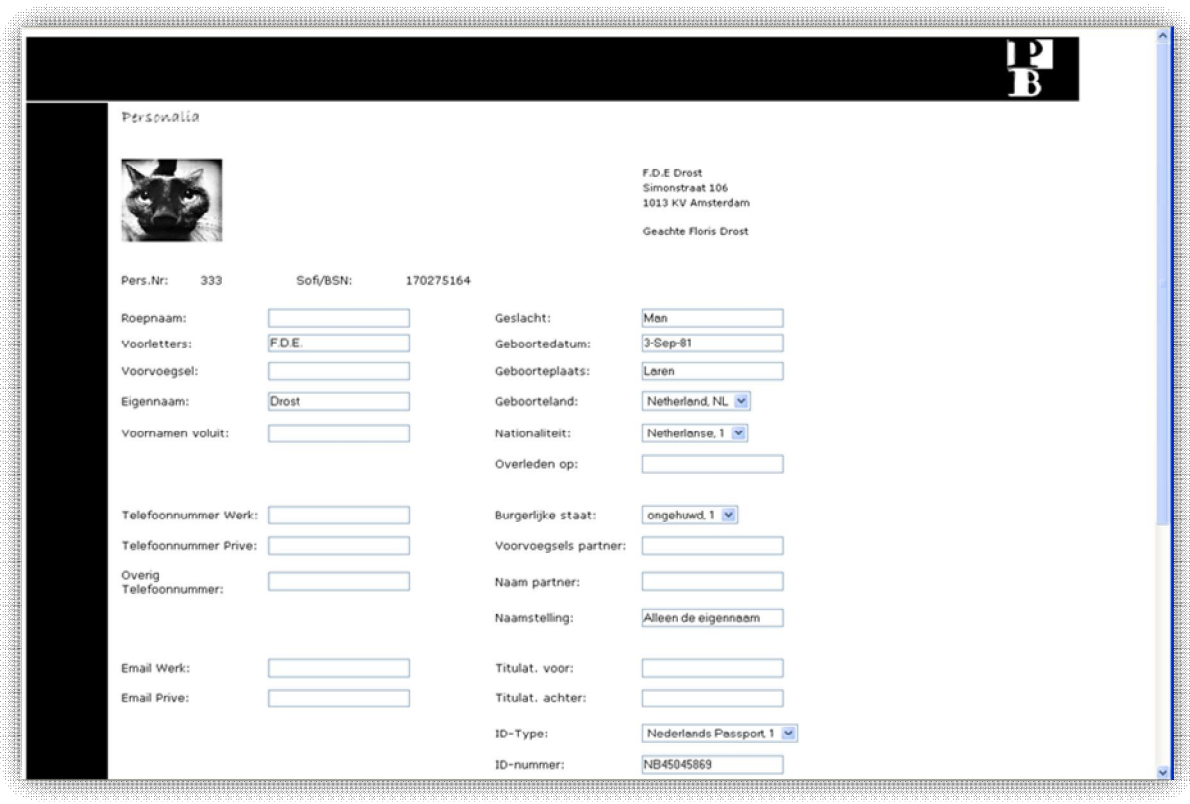


Figure 24: First prototype evaluation (Scrolling)

– The historic information is always visible.

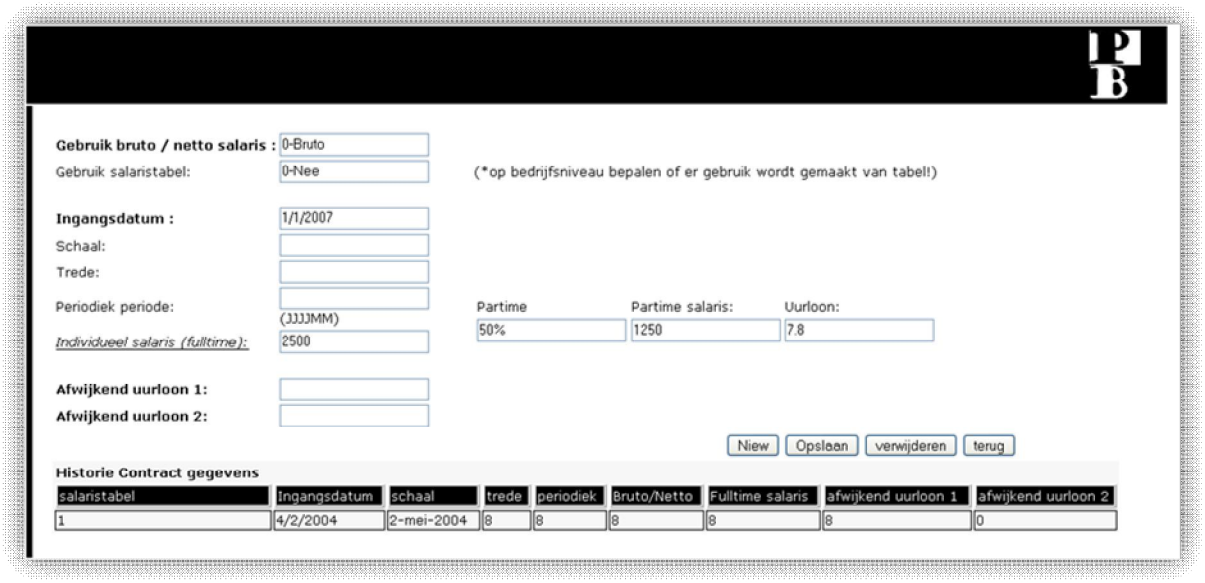


Figure 25: First prototype evaluation (Historic)

Appendix B.2 *Second Prototype*

Navigation

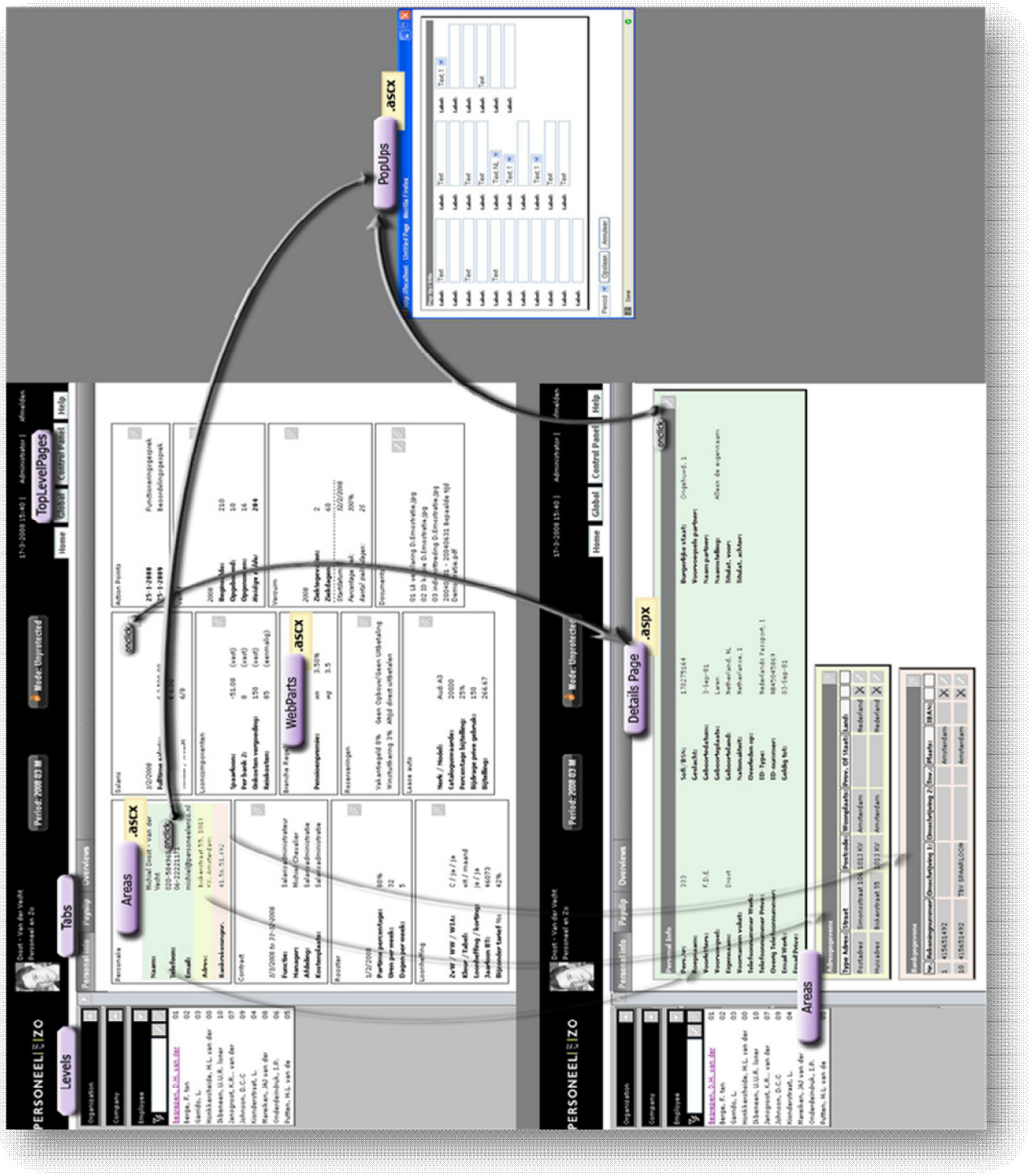


Figure 26: Home Module

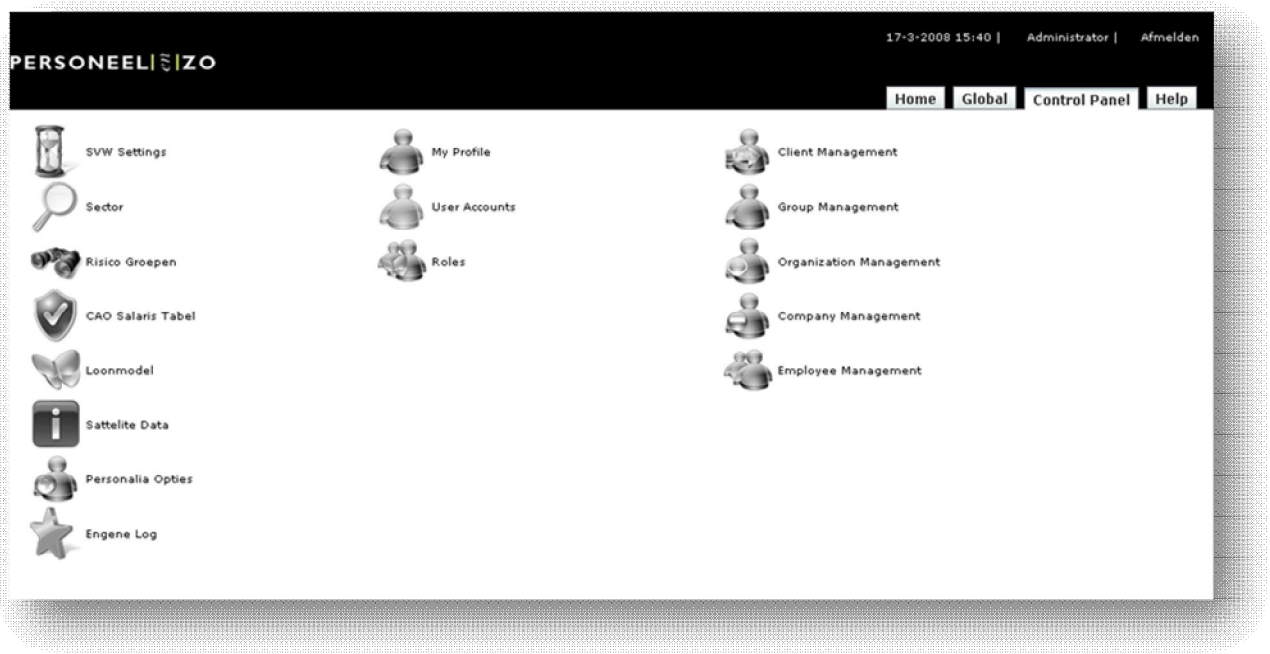


Figure 27: Control Panel Module

Screen types

Detail page areas: These are the four screen types that will be applied on the detail page areas. On them, fixed information is shown and the user will be able to carry out some actions as edit, make new or open the historic. All these actions will be triggered by clicking on one button, and a popup will appear. The popup screen types are shown on the next section.

Screen Type d1: Represents a form with fixed information on it and just one button to edit this information. A popup *screen type p1* or *screen type p2* will popup.
This type of areas can be found for Personal Information, Loonheffing, Sociale wetten, Ziekengeld and CBS detail areas between others.

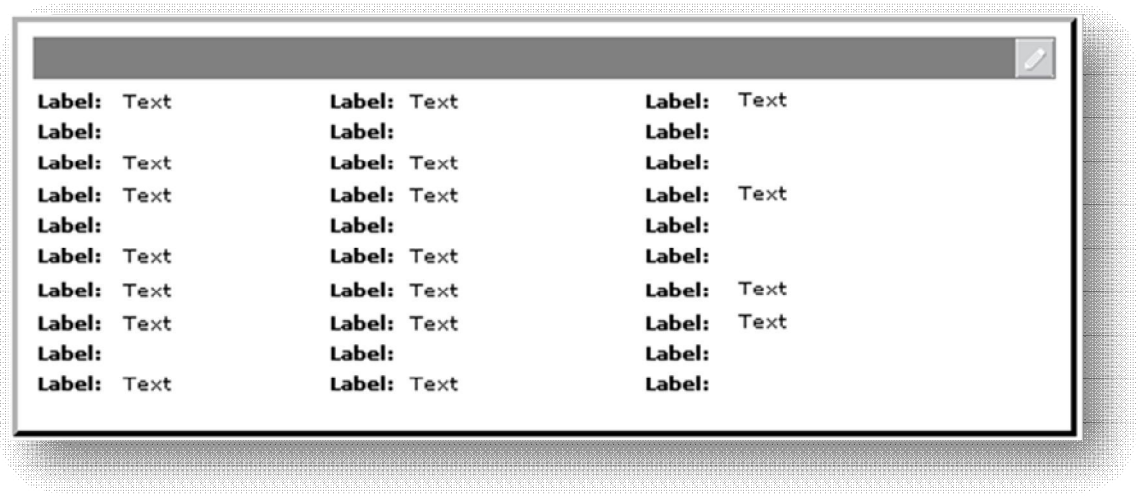


Figure 28: Screen type d1

Screen type d2: Represents a form with fixed information on it and three buttons to edit or remove this information, and open the Historic popup. A popup *screen type p1* or *screen type p2* will popup clicking on the *edit* button. Popup *screen type p3* will be for the *historic* one. The remove button will remove the current information, which is the first row of the historic table.
This type of areas can be found for Salary, Contract, Rooster and Lease auto detail areas between others.

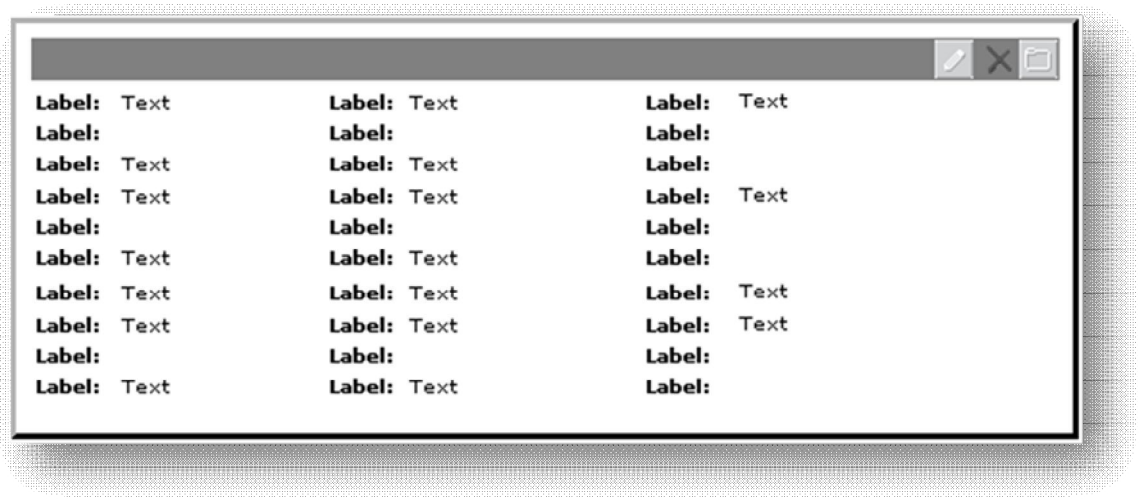


Figure 29: Screen type d2

Screen type d3: Represents a table with fixed information on it. The user could delete and edit every table row. He will have the possibility of making a new one as well. A popup *screen type p1* or *screen type p2* will popup clicking on the *edit* or *new* button.
This type of areas can be found for Adresgegevens, Bankgegevens, Action Points, Verlof, and Verzuim detail areas between others.

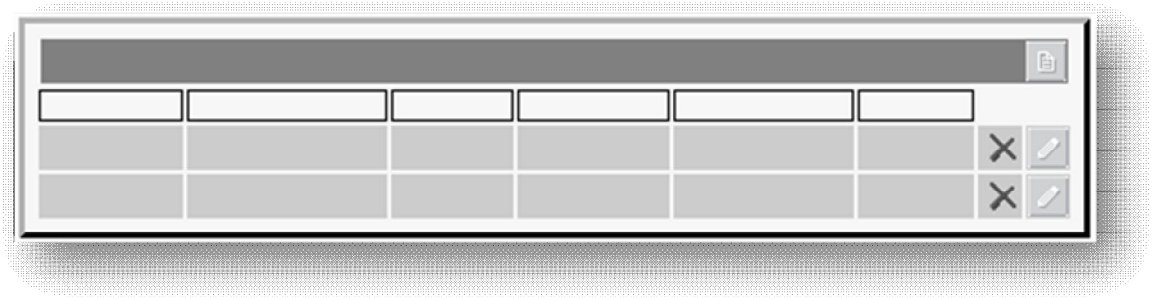


Figure 30: Screen type d3

Screen type d4: Represents a table with fixed information on it. The user could delete every table row. He will have the possibility of making a new one or see the historic as well. A popup *screen type p1* or *screen type p2* will popup clicking on the *edit* or *new* button. Popup *screen type p3* will be for the *historic* one.
This type of areas can be found for Looncomponenten detail areas between others.

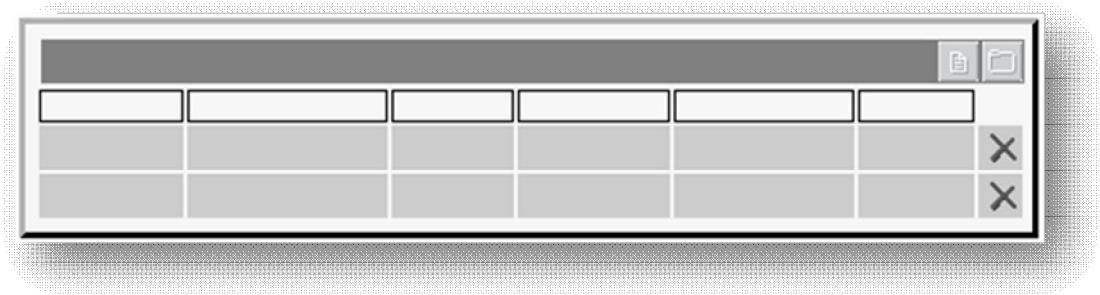


Figure 31: Screen type d4

Popups: These are the three screen types applied for the popups. All of them have the Save and Cancel buttons in common. The information on them is editable. This one can be the information that was already in a form and the user wants to edit, or default values when the user wants to create a new entrance.
To save the changes the user will have to click on Save, and by clicking on Cancel nothing will be saved. No matter the button the user press, the popup will be always closed.

Screen type p1: This screen type can be found for Salary, Action Points, Contract, Looncomponenten, Verlof, Rooster and Verzuim popups between others.

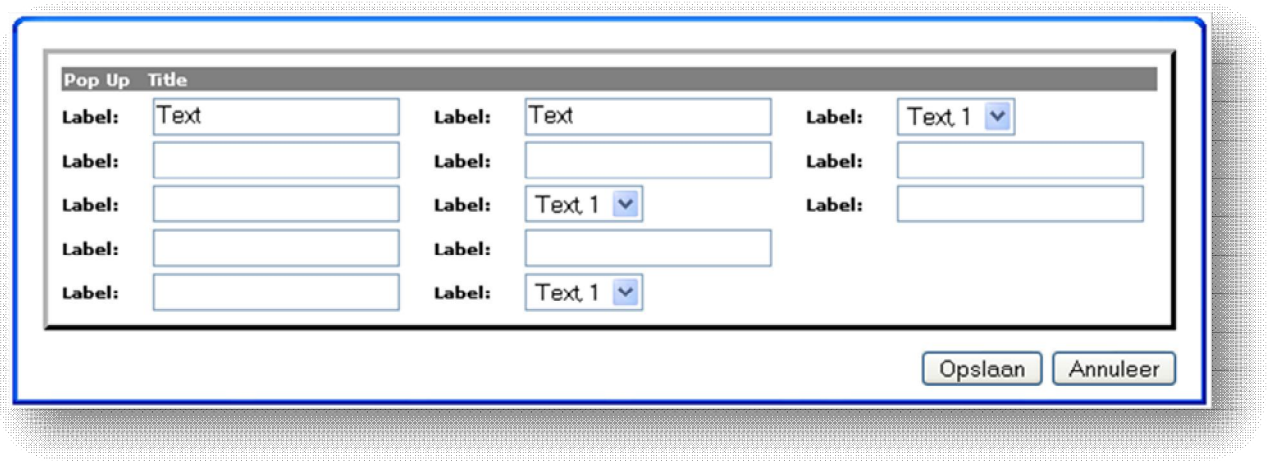


Figure 32: Screen type p1

Screen type p2: This kind of popup includes a Period picker. All in this application is based on a certain period and sometimes the user needs to specify the period in which this information will be applied.
This screen type can be found for PersonalInfo, Address, Bank Account, Reserveringen and Loonheffing popups between others.

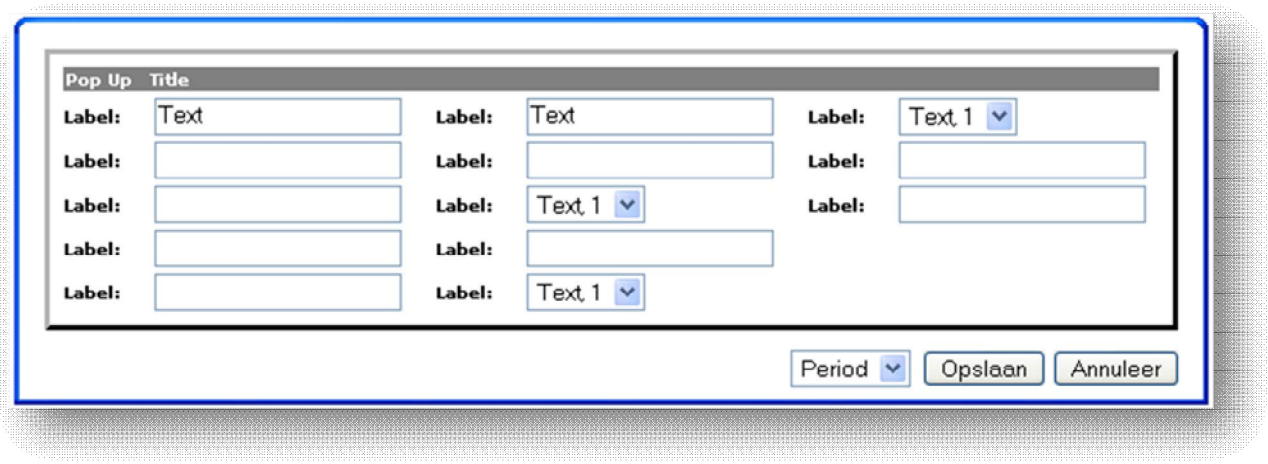


Figure 33: Screen type p2

Screen type p3: This screen type is mostly applied for the historic view.

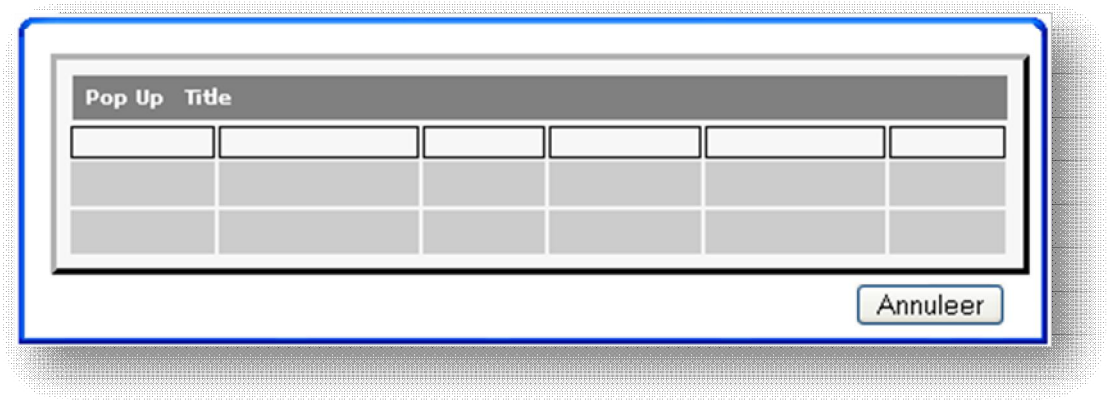


Figure 34: Screen type p3

Technical features

Retractable Menu

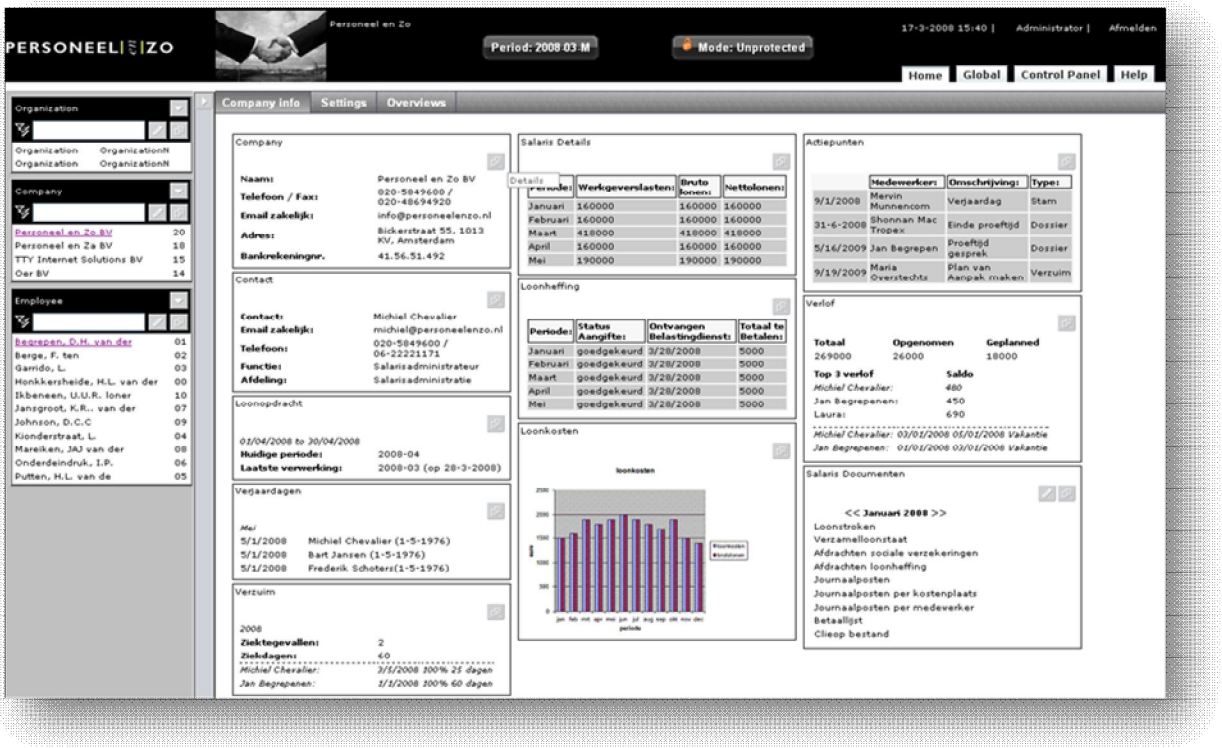


Figure 35: Left menu opened

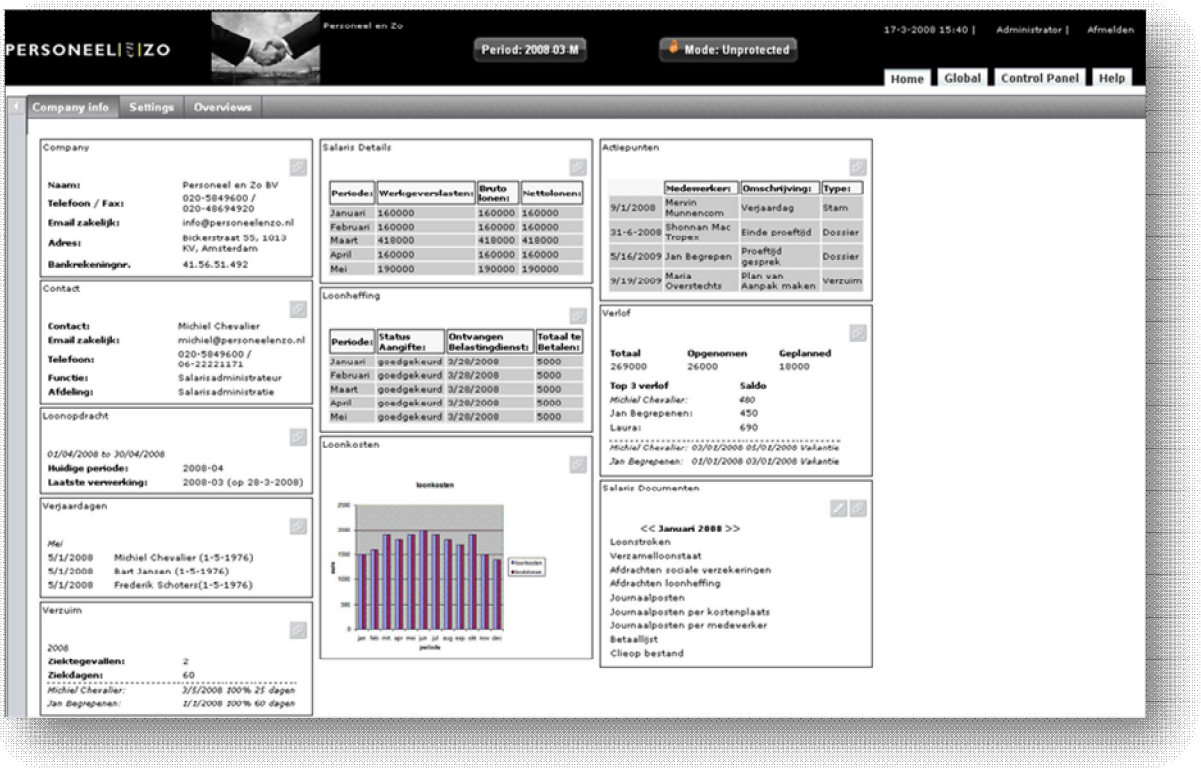
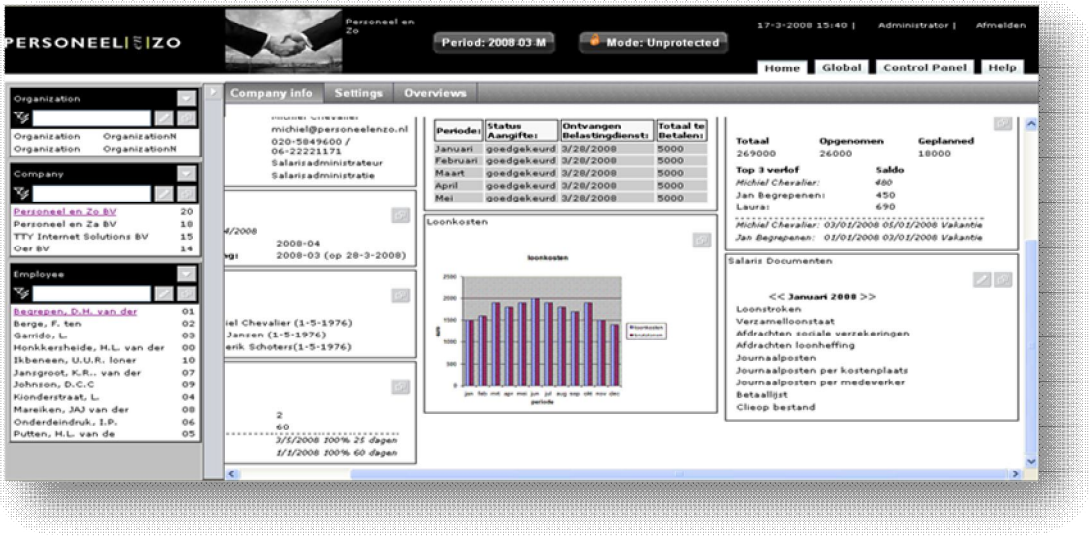


Figure 36: Left menu collapsed

Main content scrolling



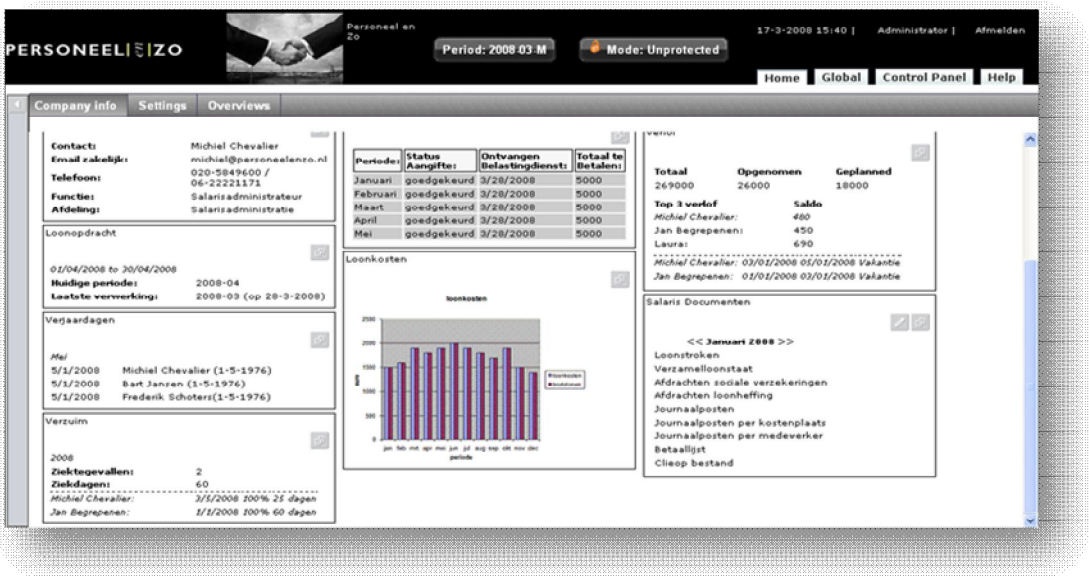


Figure 37: Main content scrolling

Evaluation

Header

Several sketches about how the header could be were made until the best one came up:

- The image was removed, and the period picker and system mode selector placed on the tab bar:

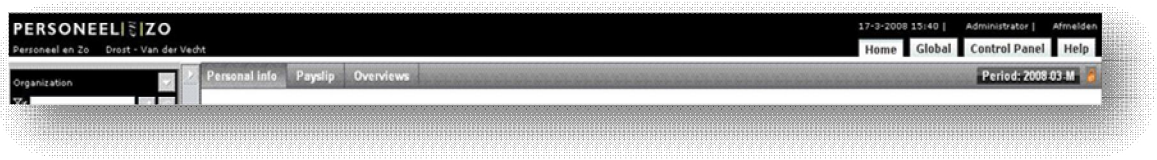


Figure 38: Header evaluation (1)

It was not a good solution because the tab bar is only going to appear for now on the Home Module and the period picker and system mode selector may be needed on other modules.

- The period picker and system mode selector were placed on the header again, and the idea of showing the employee photo on the *Personal Information dashlet* came up.

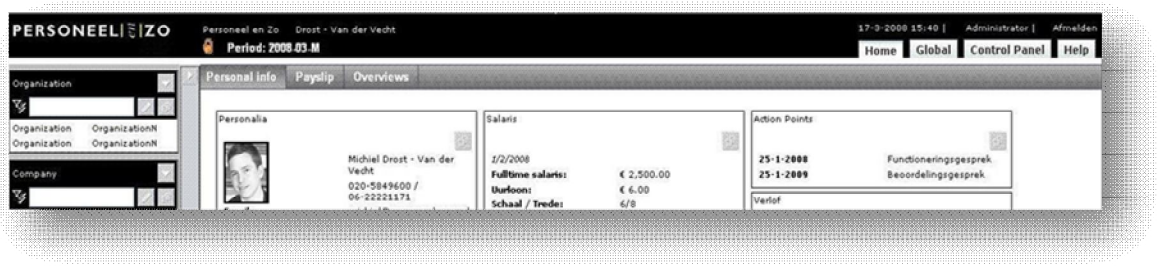


Figure 39: Header evaluation (2)

The previous model seems to work because with the employee and company names on the header and the employee photo on the dashlet, the user cannot forget about with whom he is working and at the same time the header becomes smaller and more consistent. But a small change is done, removing the label *Period* and just using a lock icon for the system mode.

Figure 40: Header evaluation (3)

Dashlets

A header is added on every dashlet which emphasizes more what is the dashlet about and also lets us to define a place which the user can use to drag the dashlet around. On this header the close button and the one to collapse the dashlet content are going to be placed.

It was difficult to reach the icon on a dashlet to go into the detail page. So it was replaced by a bar at the bottom of the dashlet.

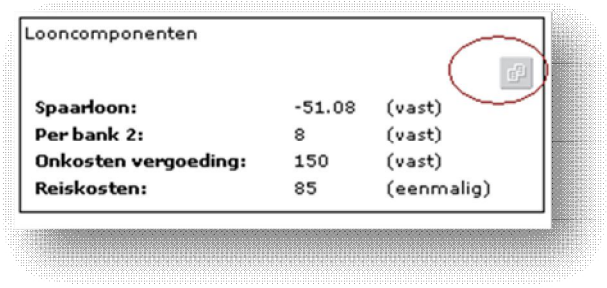


Figure 41: Dashlets before evaluation

Figure 42: Dashlets after evaluation

It was thought about the way in which a dashlet, who has been removed, should be added again on the user dashboard. In a first approach, a button was placed, *add dashlet*, at the top of the dashboard just behind the tab bar. When the user wanted to add a dashlet on his dashboard, he had to click on this link and an area appeared with all the possible dashlets to add. The user should select the dashlets to add and click on a button to add them. It had two problems:

- If there are too many possible dashlets to add, this area can become too large, taking place to the main content and moving it down, what becomes a bit annoying, because when this area is closed the main content is moved up again.
- The user has to click once to select the dashlet, and another time to add it into the dashboard.

A better solution is found. An icon with a plus and a tooltip, *add dashlet*, is placed on the top right corner of the main content area. When the user clicks on the icon a panel with only the necessary size is opened and the icon becomes a cross. This way as many possible dashlets there are bigger the panel will be. As it is a panel which overlaps the main content and it is on the right side, it doesn't disturb the user at all.
Only the dashlets which are not on the dashboard will appear on this panel. And just by clicking on a dashlet name, it will appear on the dashboard and disappear from the panel to add.

Figure 43: Catalog closed

Figure 44: Catalog opened

Left Menu

The handle to put aside the left menu is not necessary to be so width. It is reduced into the half and it becomes nicer and still usable.

As the space needs to be optimized, the panels on the left menu need to be resizable.

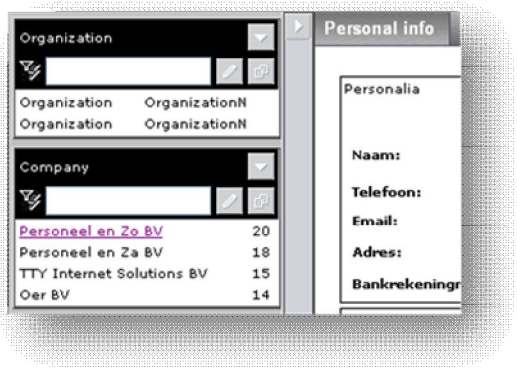


Figure 45: Left Menu before evaluation

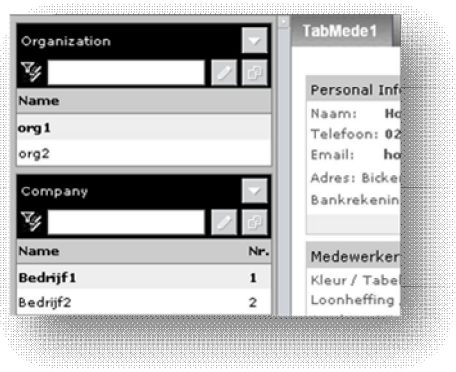


Figure 46: Left Menu after evaluation

Historic

The way the historic is working is changed. The delete button on a detail page removes the detail page information, showing the previews one. And here there is a problem. It was meant to delete the first row of the historic, and as it is possible to change information in the past or in the future, the information shown on the detail page might not be the one that the user wants to remove (which is the last one he created). This will make the user get confuse.

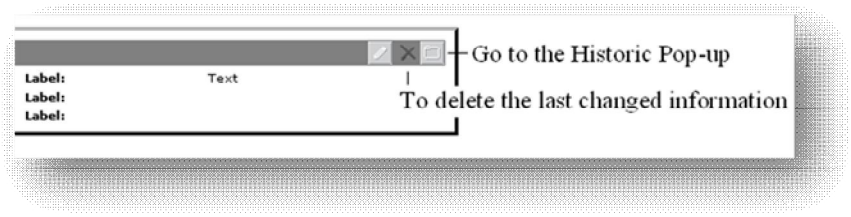


Figure 47: Historic before evaluation

Now on the detail page is going to appear just a button to go to the historic popup. And this one is going to work as a real historic in which you can delete the last change made. For that purpose, the delete button is no longer going to be on the detail page but on the first row of the historic.

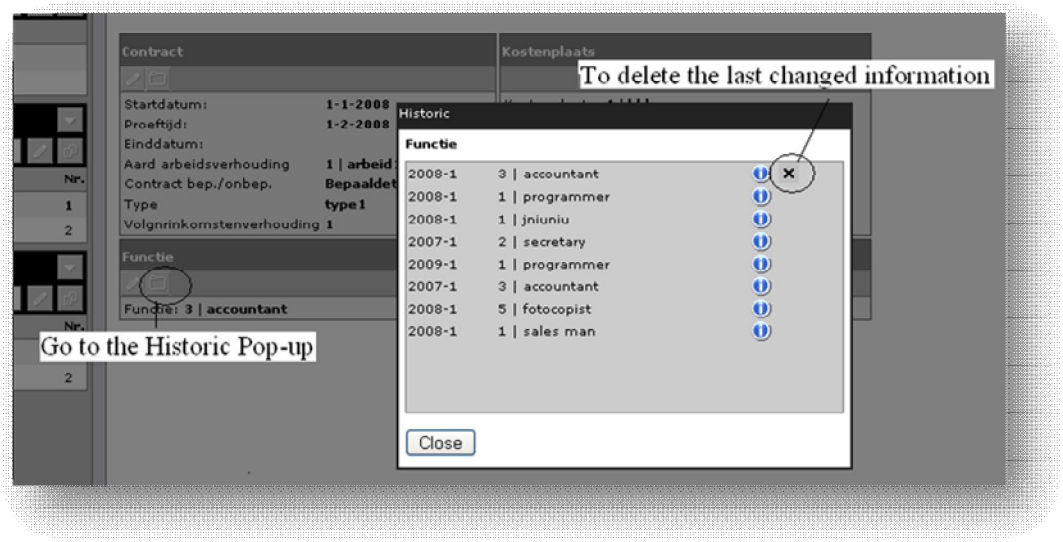


Figure 48: Historic after evaluation

The user is also going to be able to click on each row of the historic in which the row information is going to be shown in detail. When the user placed the mouse over the "i" icon, it will be shown information about who created that entrance and when.

Appendix C. Technical Design

Appendix C.1 Dashboard

Dashboard Control

```
<div id="catalog">
	<div id="catalog_buttons">
		<img class="pointed_img" src="/close.gif" style="float: right;" onclick="CloseCatalog();"/>
	</div>
	<div id="catalog_list">
	</div>
</div>

<div id="btnCatalogOpen">
	<div>
		<img class="pointed_img" src="/add.gif" style="float: right;"
			onclick="OpenCatalog();"
			onmouseover="javascript:return ToolTip('Add Dashlet', 0);"
			onmouseout="return nd();"/>
	</div>
</div>

<br />

<table style="height: 500px;">
	<tr>
		<td style="vertical-align: top;">
			<ul class="navigation" id="zone_1" style="height: 350px; width: 300px;"></ul>
		 </td>
		<td style="vertical-align: top;">
			<ul class="navigation" id="zone_2" style="height: 350px; width: 300px;"></ul>
		</td>
		<td style="vertical-align: top;">
			<ul class="navigation" id="zone_3" style="height: 350px; width: 300px;">< ul>
		</td>
	</tr>
</table>
```

LoadDashboard ()

```
function LoadDashboard ()
{
	var strReceiveFunc = "";
	strReceiveFunc += "eval('webpartsArray =' + httpRequest.responseText);";
	strReceiveFunc += "intWebPartsTotal = getWebPartsCount(webpartsArray);";
	strReceiveFunc += "LoadZones (webpartsArray);";

	sendAJAX3("+handleLoadDashboard ,
			'intNavNodeID='+intNavNodeID+' & intPersonID='+intPersonID,
			strReceiveFunc);
}
```

LoadZones (list)

```
function LoadZones (zonesArray)
{
	for (i=0; I < zonesArray.length; i++)
	{
		for (j=0; j < zonesArray[i].length; j++)
		{
			LoadWebPart (zonesArray[i][j], i+1);
		}
	}
}
```

LoadWebPart (Id, Zone)

```
function LoadWebPart (intWebPartID, intZone)
{
	if (intWebPartID != '0')
	{
		var zone = document.getElementById("zone_" + intZone);
		var newWebPart = document.createElement('li');

		newWebPart.setAttribute ('id','item_'+  intWebPartID);
		newWebPart.setAttribute ('style','position: relative;');
		newWebPart.innerHTML ="<div style ="+styleLoadWebPart+" <img src='/
                                                webpart_loading.gif'/></div>";
		zone.appendChild (newWebPart);

		var strReceiveFunc = "";
		strReceiveFunc += "var cmd_content = httpRequest.responseText.split('$');";
		strReceiveFunc += "var webpart = document.getElementById (\"item_\" + cmd_content[0]);";
		strReceiveFunc += "webpart.innerHTML = cmd_content[1];";
		strReceiveFunc += "intWebPartsLoaded++;";
		strReceiveFunc += "CreateSortables();";

		var args = 'intNavNodeID='+intNavNodeID+ &
			intPersonID='+intPersonID+' &
			intWebPartID='+intWebPartID;

		sendAJAX3(handleLoadWebPart, args, strReceiveFunc);
	}
}
```

Appendix C.1 Popups

Popup Control

```
<div id="div_popup_bg">
</div>

<div id="div_popup_window">
        <div id="div_popup_header" onmousedown="dragStart(event, 'div_popup_window')">
        </div>
        <div id="div_popup">
        </div>
        <div id="div_popup_buttons">
                <asp:Button ID="CancelButton" OnClientClick="ClosePopup(); return false;" runat="server"
                Text="Close" />
        </div>
</div>
```

OpenPopup (id, args)

```
function OpenPopup (popupid, args)
{
        var div_popup_window = document.getElementById("div_popup_window");
        var div_popup_bg = document.getElementById("div_popup_bg");
        var div_popup_header = document.getElementById("div_popup_header");

        setOpacity(div_popup_window, 0);
        div_popup_header.innerHTML = name;
        div_popup_bg.style.visibility = "visible";

        var func = "CallServerPopup('LoadPopup$" + popupid + "$" + args + "', ')";
        eval(func);
}
```

ClosePopup (id, args)

```
function ClosePopup()
{
        var div_popup_window = document.getElementById("div_popup_window");
        new Effect.Opacity ('div_popup_window',
                        { fps : 50,  duration : intOpacityEffectDuration, from: 1, to: 0 ,
                                afterFinish: OnAfterClosePopup});
}
```

OnAfterClosePopup

```
function OnAfterClosePopup()
{
	var div_popup = document.getElementById("div_popup");
	var div_popup_window = document.getElementById("div_popup_window");
	var div_popup_bg = document.getElementById("div_popup_bg");
	var div_popup_header = document.getElementById("div_popup_header");
	var div_popup_buttons = document.getElementById("div_popup_buttons");

	div_popup_header.innerHTML = "";
	div_popup.innerHTML = "";

	div_popup_window.style.visibility = "hidden";
	div_popup_bg.style.visibility = "hidden";
	div_popup_buttons.style.display = "none";
	div_popup_window.style.width = widthPopUp_defaultWindow+"px";
	div_popup_window.style.height = heightPopUp_defaultWindow+"px";
}
```

References

[Caroll & Rosson 1990]
Usability Engineering - scenario-based development of human-computer interaction - Mary Beth Rosson & John M. Carroll - Morgan Kaufmann publishers

[Dumas and Redish, 1999]
A Practical Guide to Usability Testing. Joseph S. Dumas, Janice C. Redish

[James Hom, 1998]
The Usability Methods Toolbox. James Hom, 1998 (http://jthom.best.vwh.net/usability/)

[J. Nielsen, 1993]
J. Nielsen "Usability Engineering", pp.199-200, Academic Press, 1993.

[John Wiley & Sons]
Rubin, J.Handbook of Usability Testing. John Wiley & Sons.

[H. Rex Hartson]
H. Rex Hartson et. al., Remote Evaluation: The Network as an Extension of the Usability Laboratory

[Lindgaard, 1994]
Lindgaard, G., Usability Testing and System Evaluation: A Guide for Designing Useful Computer Systems, 1994, Chapman and Hall, London, U.K. ISBN 0-412-46100-5

[Nielsen, Jakob]
Nielsen, Jakob, "Guerrilla HCI: Using Discount Usability Engineering to Penetrate the Intimidation Barrier," online Web page at http://www.useit.com/papers/guerrilla_hci.html.

[Rubin, Jeffrey]
Rubin, Jeffrey, Handbook of Usability Testing, 1994, John Wiley and Sons, New York, NY
ISBN 0-471-59403-2 (paper)

[C. Wharton]
C. Wharton et. al. "The cognitive walkthrough method: a practitioner's guide" in J. Nielsen & R. Mack "Usability Inspection Methods" pp. 105-140

[Vicki L. Sauter, 1999]
http://www.umsl.edu/~sauterv/analysis/prototyping/proto.html

[GUUUI]
http://www.guuui.com/issues/01_03_02.php

Other references:

- New Web-Usability Evaluation Method : Scenario-Based Walkthrough. Satoko Segawa, Masahico Sugimura, Kazushi Ishigaki, October 15, 2004 (http://www.fujitsu.com/downloads/MAG/vol41-1/paper14.pdf)
- Interaction Design - beyond human-computer interaction - Yvonne Rogers, Helen Sharp & Jenny Prece - John Wiley & Sons, Inc.
- Beginning.ASP.NET.2.0.pdf
- http://asp.net
- http://www.usability.gov
- http://www.welie.com
- http://developers.sun.com/docs/web-app-guidelines/uispec4_0/index.html#toc

- http://www.asp.net/ajax/ajaxcontroltoolkit/samples/
- http://whatis.techtarget.com
- http://msdn.microsoft.com/en-us/default.aspx
- http://script.aculo.us/