Karlsruher Institut für Technologie
Institut für Arbeitswissenschaft und
Betriebsorganisation
o. Prof. Dr.-Ing. Dipl.-Wirtsch.-Ing. Gert Zülch

Altes Maschinenbaugebäude
Kaiserstr. 12, 76131 Karlsruhe
Telefon (0721) 608-44250
Telefax (0721) 608-47935
E-Mail info@ifab.kit.edu

cand. mach. Santiago Boix Lago
Matr.-Nr. 14380400

Modeling of Working Time Models and Rosters

Diplomarbeit
(Master Thesis)
-

Betreuer:	o. Prof. Dr.-Ing. Dipl.-Wirtsch.-Ing. Gert Zülch
Mitbetreuender wiss. Mitarbeiter:	Dipl.-Inform. Michael Leupold

Karlsruhe, 06.06.2012

Karlsruher Institut für Technologie
Institut für Arbeitswissenschaft und Betriebsorganisation
o. Prof. Dr.-Ing. Dipl.-Wirtsch.-Ing. Gert Zülch

Altes Maschinenbaugebäude
Kaiserstr. 12, 76131 Karlsruhe
Telefon (0721) 608-44250
Telefax (0721) 608-47935
E-Mail info@ifab.kit.edu

[Aufgabenstellung wird hier unterschrieben mit eingebunden. 2 Exemplare mit einer Kopie, 1 Exemplar mit der Originalversion der Aufgabenstellung (die gibt es kurz vor Fertigstellung im Sekretariat bei Frau Schlund)]

Hinweise:
Es handelt sich hierbei nur um eine Rohfassung einer Dokumentvorlage. Das bedeutet hauptsächlich:

- Die Vorlage muss entsprechend ergänzt werden. Sinnvolle Ergänzungen übernehme ich natürlich sehr gerne in meinen Master, um ihn nachfolgenden Studierenden zur Verfügung zu stellen.
- Die Vorlage hat keinen Anspruch auf Übereinstimmung mit den Richtlinien, auch wenn darauf bei der Erstellung natürlich geachtet wurde. Also alles ohne Gewähr. Bindend ist aber auf jeden Fall das bei Beginn der Arbeit ausgeteilte Merkblatt. Vor Abgabe sollte jeweils überprüft werden, ob das Endprodukt den Anforderungen dieses Merkblatts entspricht.

Abschnitt 3.2 des Merkblatts spricht von Verzeichnissen, die man bei Bedarf einbauen kann. Generell ist so ein Verzeichnis ganz praktisch, z.B. wenn man eine größere Zahl von Abbildungen oder Formeln hat. Verschiedene Arten von Verzeichnissen sind z.B.:

- Abbildungsverzeichnis
- Formelverzeichnis
- Abkürzungsverzeichnis
- Verzeichnis verwendeter (oder selbst eingeführter) Formelzeichen.

Ob ein Verzeichnis verwendet werden sollte, kann aber im Einzelfall mit mir besprochen werden. Ich gebe gerne Tipps.

Dipl.-Inform. Michael Leupold

Karlsruher Institut für Technologie
Institut für Arbeitswissenschaft und

Betriebsorganisation
o. Prof. Dr.-Ing. Dipl.-Wirtsch.-Ing. Gert Zülch

Altes Maschinenbaugebäude
Kaiserstr. 12, 76131 Karlsruhe
Telefon (0721) 608-44250

Telefax (0721) 608-47935
E-Mail info@ifab.kit.edu

Karlsruher Institut für Technologie
Institut für Arbeitswissenschaft und Betriebsorganisation
o. Prof. Dr.-Ing. Dipl.-Wirtsch.-Ing. Gert Zülch

Altes Maschinenbaugebäude
Kaiserstr. 12, 76131 Karlsruhe
Telefon (0721) 608-44250
Telefax (0721) 608-47935
E-Mail info@ifab.kit.edu

Erklärung zur Diplomarbeit

Ich versichere hiermit, dass ich die Arbeit mit dem Thema

Modellierung von Arbeitszeitmodellen und Einsatzzeiten
(Modelling of Working Time Models and Rosters)

selbständig angefertigt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie die wörtlich und inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Regeln zur Sicherung guter wissenschaftlicher Praxis im Karlsruher Institut für Technologie (KIT) in der jeweils gültigen Fassung beachtet habe.

Karlsruhe, 06.06.2012 _____

Ort & Datum Unterschrift

Karlsruher Institut für Technologie
Institut für Arbeitswissenschaft und Betriebsorganisation
o. Prof. Dr.-Ing. Dipl.-Wirtsch.-Ing. Gert Zülch

Altes Maschinenbaugebäude
Kaiserstr. 12, 76131 Karlsruhe
Telefon (0721) 608-44250
Telefax (0721) 608-47935
E-Mail info@ifab.kit.edu

Acknowledgements

I would like to thank Dipl.-Inform. Michael Leupold and Prof. Dr.-Ing. Dipl.-Wirtsch.-Ing. Gert Zülch for giving me the possibility to write my diploma thesis at the Institute für Arbeitswissenschaft und Betrieborganisation for his help and good supervision. Without the work of any of them, the development of this project would have been impossible.

I would also like to thank Joel T. Zimmerman, Nick Rotteveel, Paul Kalkbrenner, David Dorad, Uroš Umek, Norman Q. Cook, Guy-Manuel de Homem-Christo, Thomas Bangalter and many others for the company in the endless „sessions“ of library. Special thanks to Manel for making my mornings happier.

I would also like to thank from the bottom of my heart all my home friends and all my Erasmus friends who made me feeling in Karlsruhe like at home. I would also like to thank my family for bearing me up all these years, specially the lasts months. This work would not be finished without all them.

Karlsruhe, 06.06.2012

Ort & Datum

Unterschrift

Table of Contents

Illustrations Index

Index of Tables

Abreviations used:

CISQ: Consortium for IT Software Quality

IEEE: Institute of Electrical and Electronics Engineers.

IFAB: Institut für Arbeitswissenschaft und Betriebsorganisation (Institute of Human and Industrial Engineering)

ILO: International Labour Office

ISO: International Organization for Standardization

OMG: Object Management Group

TOIL: Time off in Lieu

UML: Unified Modeling Language

XML: eXtensible Markup Language

1. Introduction

A couple of terms will be clarified before starting. A working time model is an arrangement during which an employee works a certain amount of time for an employer in exchange of something (usually money). A roster is basically a list, especially of names but in this project a roster will be a list of workers' names and the times when they are required to work (organized as a table). The words schedule and timetable will be sometimes used as a synonym of roster in this paper.

1.1 Objectives

In order to enter data and process working time on a computer and assign slots in the roster of employees, an appropriate data model and input-/display procedures are essential. The main objective of this project is to develop a data model for working time models and rosters and evaluate it. Therefore it is important to take into account the different actual working time models. The according tools and interfaces should be designed to support the user of the system instead of constraining him.

Therefore, the goals of this Master Thesis are:

- To give an overview of the working time models existing in the labor market and the stakeholders.
- To review literature and software in the field of data models and input-/display methods for working time models and rosters in order to build up a data model and an application.
- To implement and to evaluate our application. This application is also compared to other existing software.

1.2 Scope

The main aim of this Master Thesis is to show that it makes sense to study the working time models field in order to try to improve or to find a proper data model. After this analysis is done, it will be decided if the final data model is going to be implemented or if any of the other mentioned models are going to be further developed. Of course an existing data model can only be expanded if the author allow it.

The following tasks will be completed:

- Review of literature and software in the field of data models and input-/display methods for working time models and rosters.
- Inventory of working time models

- Development of a data model for the working time models
- Implementation of this data model (with a proper Data Base and User Interface)
- Evaluation of the application and comparison with other software solutions.

This thesis will try to take into account all the working time models. This will be complicated since there are many working time models. Our target will be to analyze as many working time models as possible. This set should be seen as a summarized time models group and any existing time model should be able to be assimilated into one of them.

It is also important to say that some working time models will be studied and mentioned during the whole diploma thesis, but they will not be modeled, as these time models can be combined with all the working time models studied. For example, a phased retirement can also be seen as a voluntary reduced working time or even as a sabbatical period.

1.3 Justification

1.3.1 Times are changing

RIFKIN (2011) in his book "The third industrial revolution" explains that our industrial civilization is at a crossroads. Oil and the other fossil fuel energies that make up the industrial way of life are sunsetting, and the technologies made from and propelled by these energies are antiquated. The entire industrial infrastructure built off of fossil fuels is aging and in disrepair. The result is that unemployment is rising to dangerous levels all over the world. Governments, businesses and consumers are awash in debt and living standards are declining everywhere.

Since the beginning of the Great Recession in the summer of 2008, governments, the business community, and civil society have been embroiled in a fierce debate over how to restart the global economy. Finding that new vision requires an understanding of the technological forces that precipitate the profound transformations in society.

"With the more fierce competition of enterprises, low cost strategy is altering the life of people. Employees are working longer hours, and in some cases have more duties to perform, which increases work demand. Women are entering the workforce at an increasing rate, many have children or dependents. Further, in many households both partners work, creating dual demands in work and family for both individuals. Physical and psychological health, such work stress, job

burnout, marital crisis, force individuals change their work philosophy to pursue the balance between work and personal life outside of work" (MA, XU 2009 p. 1).

1.3.2 New panorama

New working time models like flexible time emerged in the past years thanks to the new technologies. According to LEE, MCCANN and MESSENGER (2007, p. 120), companies and workers all around the world tend in the same direction: more flexibility. This recent trend has made necessary for businesses to introduce their own working programs and to reduce the risk of losing valued employees. At the same time, flexible options not only strengthen commitment, but also give employees more time to handle the very situations that sometimes lead to absenteeism.

The main goal of this project is to understand the existing working time models and try to give an overview of them. Once this part is done, a data model of the working time models will be established (or improved from an existing one) and implemented (if not already done). There are dozens of different applications which already deal with working time models and rosters. These already give us an idea of how the market actually is. Despite the high number of available applications, no one allows the user to directly choose a concrete working time model in order to prepare working schedules.

1.4 Outline

The thesis is divided in seven chapters, one per each step of work.

Chapter 1 is an introduction which contains the main objectives. The limits of the project are clearly established.

In chapter 2, the so called "Context" of the project will be presented. This chapter also contains a short overview of working time history, the private lives of workers and scheduling problems. Then a review of working time models, rosters and commercial software is presented. This chapter will be important to understand the steps to follow during this thesis.

Chapter 3 will contain an inventory of the most important working time models. The purpose of this section is to know and understand the most relevant working time models that are going to be modeled and implemented in this project.

In chapter 4, the description of the UML modeling of the working time models will be made. After a quick introduction to UML, an explaination of all the steps from the first conception of the model until the last version will be presented. The way that concrete time models are modeled will also be explained.

Chapter 5 is the section of this project where the implementation of the model will be analyzed. It will contain three differentiated parts: first, the system; then, the input and output interface and finally, the data base. The deployment of the software will also be commented in this part of the project.

Chapter 6 will be the evaluation chapter. It will include a qualitative survey with volunteers who will try our software.

Finally in chapter 7, the conclusions and the future prospects will be made, with explanations of what can be improved. This chapter will also contain the summary of the project.

2. Context

In order to understand the whole concept of the project, the main goals of the project will be discussed as well as the history of working time models focusing on the conflict between working time and private life (and the problems arising from this). An overview of working time models history and trends is provided in the second part of this chapter. Finally, some commercial software reviews will be presented

2.1 Working time conflict and history

2.1.1 Origin: Industrial revolution

"In the eighteenth century, the combination of technical innovations that we known as the Industrial Revolution made the factory both feasible and dominant. From the 1770s on, then, an increasing number of workers found themselves employed at jobs that required them to appear by a set time every morning and work for a day whose duration and wage were a functions of the clock" (BLYTON 1985, p. 1). New labor habits were formed and a new time-discipline was imposed through bells and clocks.

In the 19th century, the labor movement appeared to reduce the standard hours in paid employment in order to increase hours for leisure and to prevent feared technological unemployment. There was a growing sense that the work journal was too long. This was an important problem for society which later lead to some writers trying to reflect this problem in their works. One of the most important books in this field was Germinal by Émile ZOLA (2001).

It is true that the industrial revolution marks a fundamental change in the significance of time in the work process. Words like discipline, time keeping and control of time became characteristics of an industrial system based on predictability, regularity and synchronization. Two centuries away from the industrial revolution time remains a principal component of the modern work organization and a crucial aspect of the deal struck between employer and worker. The concept of time spent at the workplace represents an essential characteristic of the nature and experience of work; overtime, part-time, time and motion, etc…

Annual and weekly hours in Europe declined in recent decades. Several social and institutional factors account for these differences between countries, including statutory and collectively negotiated restrictions on standard and overtime hours in the European countries. There has been a

renewed interest from governments and unions in reducing working time in the European Union, largely as an employment policy tool. Some countries in western Europe have cut the standard work week, sometimes even down to a 35- hour work week (PLANTENGA, REMERY 2010 p.22-23).

There is also evidence of a proliferation of jobs with nonstandard working hours and variability of scheduled hours, which represents a departure from the historical standardization. In the USA and throughout Europe especially there is an increasing diversity of work patterns across sectors, industries, and even individual workers. In the past two decades, the concept of workers' participation in decision-making has received widespread attention from both management and unions.

One work aspect in which employee influence has increased very little, however, is the pattern of working time. Any increase in worker control which participation has engendered has been concentrated on issues such as the organization of work task and aspects of working conditions, but not on the pattern of working time.

2.1.2 Working time conflict

According to BURKE (2007, p. 3), work provides many benefits such as income, social contacts, the opportunity to acquire and use skills, feelings of accomplishment as well as a sense of purpose and meaning. But work also has some costs such as fatigue, time away from friends and family, and in some cases, psychological or physical health problems. The working time conflict starts when costs surpass benefits.

Some industrialized nations legally mandate a maximum work week length between 35 and 45 hours per week, and, require 2 to 5 holiday weeks per year (EVANS, LIPPOLDT, MARIANNA 2001, p. 1-3). However, a person's actual number of work hours per week cannot fall below a certain minimum without compromising a nation's ability to produce the material standards of living to which its citizens have grown accustomed. If the work week is too short compared to that society's ideal, then the society suffers from underemployment of labor and human capital. In contrast, a work week that is too long will result in more material goods at the cost of stress and other health problems. Furthermore, children are likely to receive less attention from busy parents, and their childhoods are likely to be subjectively worse. The exact ways in which long work weeks affect culture, public health, and education are debated.

Due to these changing circumstances, the International Labour Office (ILO 2008, p. 9-11) describes "Decent Work" like the sum of the aspirations of people in their working lives. It involves opportunities for work that is productive and delivers a fair income, security in the workplace and social protection for families, better prospects for personal development and social integration, freedom for people to express their concerns, organize and participate in the decisions that affect their lives and equality of opportunity and treatment for all women and men.

It is clear that currently, the working time problem is once again one of the most noticed problems in a company. The issue is a focal point for discussions about social values, family dynamics, economic competitiveness, and the very nature of economic growth and development (SFEIR-YOUNIS 2004, p.62-65). For a better understanding of the working time problem, it would be interesting to take a look at the types of working schedules and what kinds of problems appear.

2.1.3 Flexibility

According to STOCK, ZÜLCH (2005, p. 57), in recent years flexible working time systems have been established in many service departments due to the fact that they enable the adjustment of the assignment of personnel to the fluctuating appearance of customers. The configuration of a flexible working time model has turned out to be a very complex task since, in addition to the interests of the service department (work demands) and those of the employees (workers preferences), further parameters, e.g. legal regulations or ergonomic recommendations must be considered (see them in HORNBERGER, KNAUTH 2000, pp. 25). Moreover, the design of any appropriate working time model is aggravated by some factors and the most important is the complexity that arises from the numerous alternatively or complementary applicable working time models. Approximately 10000 different working time models are in use worldwide, clearly illustrating that "the" working time model does not exist.

So which working time models and how should they be modeled? The is no right answer to this question but this project will try to approach it. To do that, I am going to try to focus on flexibility during the whole project. In other words, I will try to made a flexible model which allow the user a high degree of flexibility in the way the working time models are established in the program.

2.2 Working time modeling and schedules review

Industrial revolution marks a fundamental change in the significance of time. In the essay called Advice to a Young Tradesman, Written by an Old One (1748), Benjamin Franklin wrote "Remember that time is money". In other words, time is valuable and money is wasted when a

person's time is not used productively. Since time becomes a valuable good, the need to organize personal time appears.

With the advent of new technologies, everything is computerized. Calendars and personal schedules are present since almost always on personal computers. One of the most famous softwares about these topics are Microsoft Outlook and its direct rival, Modzilla Thunderbird. Both present very intuitive interfaces, simple design and are graphically pleasing. On the next pictures, some views of the different main windows of the programs are presented.

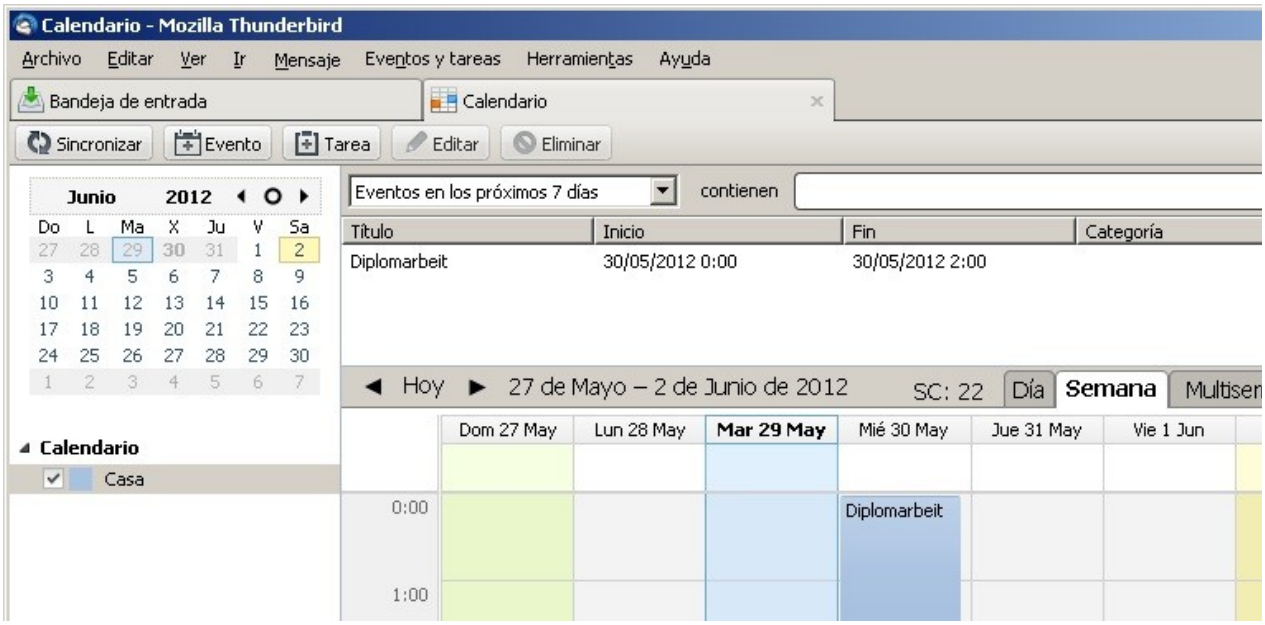


Illustration 1: Screen-shot of Modzilla Thunderbird's calendar (Spanish version)

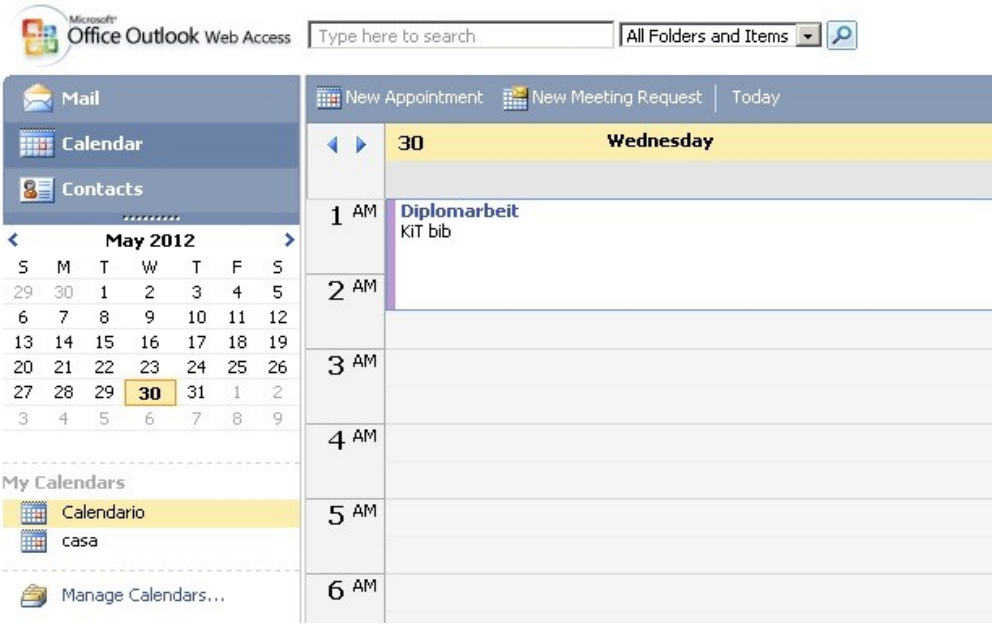


Illustration 2: Screen-shot of Microsoft Outlook's calendar (Web version)

On the picture above, a web version of the Microsoft Outlook is presented. This version is simpler than the standard version. Nevertheless, both programs are quite similar: on the left column, there is a menu with different views and a calendar for a quick search and on the center of the window the schedule for a concrete week or day. Just two examples are presented, but generally, all the software about schedules and time organization are quite similar.

On the next section, software in the domain of workday planning will be presented.

2.3 Commercial software review

One of our main objectives is the review of literature and software in the field of data models and input-/display methods for working time models and rosters.

2.3.1 E-Shift Design

The article of HÖFER, HOLZHÄUSER and LENNINGS (2009, p.17) says that the program is a simple and practical tool for those who work in the design of compliant shift patterns and ergonomic support. This excel-based program provides an easy access to the planning layer and supports the rapid and independent familiarization with the subject.

We tried to model a 3-week-shift working time in e-Shift Design during 3 weeks and the result is show below:

Illustration 3: Screen-shot from an e-Shift Design model

There is 3 different groups (A, b and C) and 3 different shifts (Frühschicht, Spätschicht and Nachtschicht) with rotations. We can see that the program is very interesting for shift design and has many different views (per week, per year, per group, …). It is also very easy to use and intuitive

and the fact that it is excel-based assures that the program is compatible with almost all OS and distributions.

On the other hand, e-Shift Design is a program focused on working time models with shifts. In fact it can only model working time models with shifts or similar.

2.3.2 Osim-GAM

"The ifab-Institute for Human and Industrial Engineering of the University of Karlsruhe (Germany) has developed a simulation-based procedure with working time models in hospitals" (ZÜLCH, STOCK, HRDINA, 2008, p. 495). "This tool allows for the simulation both of production and service companies and has already been used successfully in numerous simulation studies for configuring working time models" (ZÜLCH, STOCK, LEUPOLD, 2011, p. 2160). Working time models can be assessed in an objective, efficient and qualitative manner.

Illustration 4: Screen-shot from an OSim-GAM model

The picture above represents a 3-week-shift working time model during one week. There is 3 different groups (A, B and C) and 3 different shifts (Frühschicht, Spätschicht and Nachtschicht) with rotations. An interesting feature is that the daily shift can be allocated manually in the roster by selecting the shift on the right column and then clicking on a concrete day.

Another interesting OSim-GAM feature is the way ressources are modeled: all the elements related to work are thought as ressources. That includes workers, groups of workers, supervisors, working

environments, procedings and material consumed or employed. To assimilate all these elements to the same thing (a "ressource") simplifies the model and this should be taken into account for the development of the project.

However, in order to configure OSim-GAM properly for each concrete case, the program had to be supplemented by novel concepts. For example, according to ZÜLCH, STOCK and HRDINA (2007, p. 328), "OSim-GAM was extended through a hospital specific modeling concept". The same happened when modeling and assessing conflicts between professional and private life. I will try to find a generic model which does not need extensions for each new occurrence.

2.3.3 BASS 4

According to the official website of GAWO (2012), BASS 4 is a computer program that enables the design and evaluation of simple to highly complex work schedules. It considers in addition to a variety of legal and contract conditions, economic aspects and the actual exposure situation (in particular, scientific recommendations on working time). BASS 4 also includes work-physiological criteria, such as the number of consecutive night shifts, as well as psycho-social criteria, which include the interaction between work and leisure. A variety of useful functions and the user-friendly user interface are some of the highlights of the program. All these characteristics make BASS 4 an effective and efficient software for the design and evaluation of work schedules.

According to the article by BÖKER (2011, p. 29-32), special requirements, like high flexibility, resulting part-time work, emergency services, etc. are not shown. However, the program is able to generate relatively fast layer models, which largely meet the requirements previously entered and can be optimized manually. Usually, BASS finds a result for the input problem quickly but the generated schedule contains several errors that need to be removed manually or by changing the criteria in another generation runs. This program seems to be not very useful for modeling and scheduling simple working time models. We can see an example in the picture below:

Illustration 5: Screen-shot from a 3-shift model over 4 weeks on BASS 4 (GAWO 2010, p. 8)

2.3.4 XIMES OPA and SPA

According to XIMES (2008, p. 2-3), the company XIMES offers three different products:
Time Intelligence Solutions (TIS) – The intelligent universal tool for forecasting, determine staffing needs and personnel controlling.
Operating Hours Assistant (OPA) – Optimal staff coverage through the best possible service times.
Shift Plan Assistant (SPA) – The creative tool for flexible shift schedules

On this review, only the OPA and SPA will be analyzed. These programs support virtually every sector of the entire process, from needs assessment to specialized differentiated layer models according to BÖKER (2011, p. 28-29). The XIMES experts created an algorithm to develop their shift patterns and the various functions of the software. In addition to the program, the manufacturers offer comprehensive assistance, a hotline, counseling and training for working managers.

Ximes SPA operates in phases and allows the user to make a choice after each step to be worked with further. The generation usually takes longer than in other programs because it can be adjusted much more and has more sophisticated criteria. This program, like BASS 4 seems to be too sophisticated for simple working time models. For example, BÖKER (2011, p. 31-32) tried to create a 4-shift model and the SPA established that there will be only a few solutions that meet all criteria. The first generation result is quite acceptable and a good basis for manual optimization.

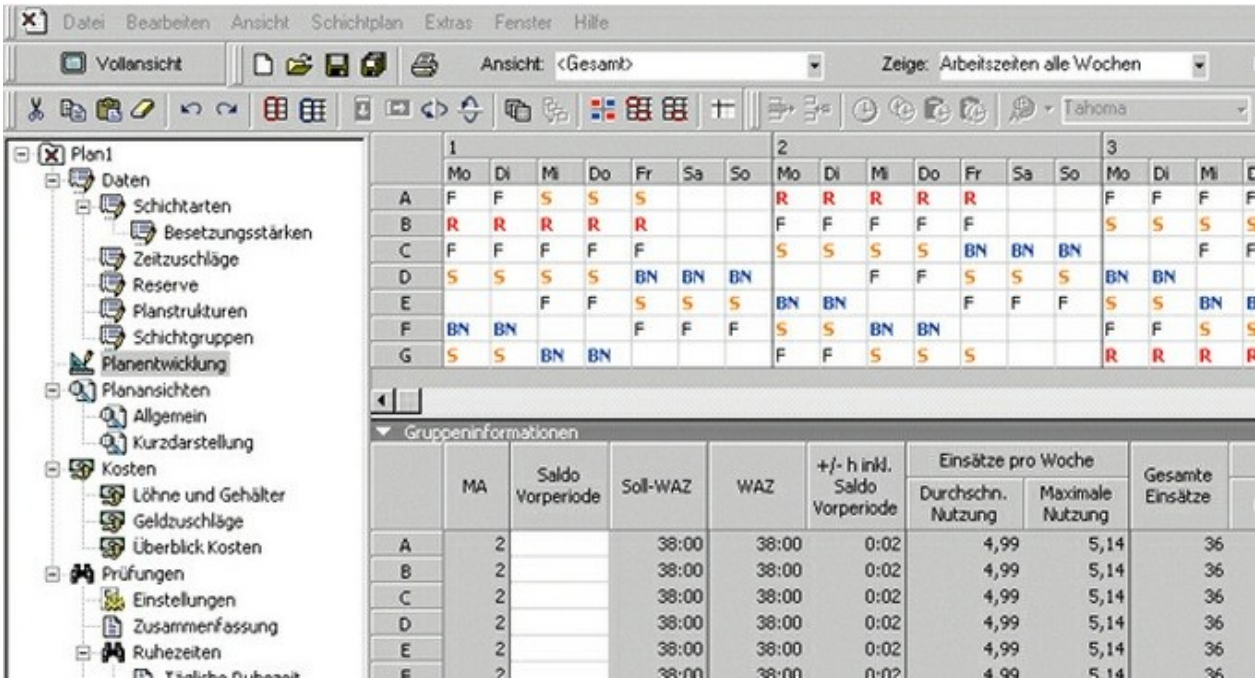


Illustration 6: Screen-shot from a 4-shift model on Shift Plan Assistant (XIMES 2012, p. 14)

2.4 Conclusions

Most of the existing software don't allow the selection of a concrete time model. This means that the user has to introduce the input data of the organization and wait to obtain a result. Usually, this output can be improved or updated with new data or filters. One of the main objectives should be to give more freedom to the user because he has the last word always.

Another interesting thing to note is that commercial software is usually very sophisticated. Although these programs are quite useful and complete, they are complicated and they are not designed for little organizations. We will also try to be realistic and focus on developing user-friendly software with full control over the working time models.

3. Inventory of working time models

Our main objective is to categorize and to draw an inventory of working time models in order to choose and to design the data model that fits the requirements of this work. For that purpose, some working time models are going to be studied emphasizing their characteristics.

There are many different working time schedules. However, it is difficult to accurately count the number of working time schedules being used by governments, trade organizations or trade unions. According to ROSA, COLLIGAN (1997, p. 6), different schedules could be used by the same occupation, the same industry or even in the same workplace adding new recombined models to the list of working time models.

So, in the first part of the chapter, the inventory of working time models will be defined, taking into account some features. All the information will be collected in order to get a final generic modeling concept for the working time models. Here comes the list of working time models that will be later modeled on the program. In other words, these working time models will be implemented in the software and available for working with them.

3.1 Full-time

Full-time employment is employment in which the employee works the full number of hours defined as such by his/her employer. Typically, the worker should start and end working at concrete hours. The difference between full-time and part-time is in the number of hours. “Forty hours is the typical number of hours people have to work to be considered a full-time employee, although working fewer hours happens” (BROWN, 2012). In fact, in some countries like Germany, that happens a lot acutually. For example, I worked as “Werkstudent” in a german company with a contract of 35 hours per week and most of my colleagues also had this kind of agreement. In any case, when trying to meet a deadline, workers work more hours generally.

Usually full-time workers are not allowed to work for other companies or to have other jobs because a full-time occupation leaves no time for anything else. That's the reason because full-time workers are more valuable to the companies. These employees participate fully in the organization, are more involved in the decision-making process and totally committed. From the worker's point of view, full-time jobs are also very interesting since “this usually means having a consistent work schedule, pay and benefits” (BROWN, 2012). In other words, usually full-time work means more stability.

3.2 Part-time

The ILO Part-Time Work Convention, 1994 (No. 175), defines a part-time worker as an "employed person whose normal hours of work are less than those of comparable full-time workers" (ILO 2004, p. 1). "Depending on the organization, part-time hours can be any number of hours under 40. A common range would be 20 to 30 hours. Some part-time employees are regular part time, while others might be contingent, such as adjunct instructors" (BROWN, 2012).

Part-time jobs have many advantages, both for the employer and for the employee. From an employer's point of view, part-time allows to "employ a person to do a specific job on a certain day or a number of days per week or a set number of hours per day. This gives the employer the flexibility to have a number of people to do specialized work within the company" (MASCARENHAS 2010). For example, let's imagine that a library needs the services of a book-keeper and a data entry operator. However, there is not enough money for paying both of them at full-time, a possible option would be to get two part-time workers specialized in those areas.

On the other side, for the employee, part-time work allows a "better balance between work and family life" (ILO 2004, p. 3). For example, "it could be that students cannot work full time and continue their studies, or mothers who wish to look after the children themselves" (MASCARENHAS 2010). Part-time jobs are also interesting because they are a very good entry point into the labor market and therefore facilitates the creation of jobs. Besides, "a part time job can also be used to supplement a full time income" (MASCARENHAS 2010).

3.3 Job-sharing

"Job sharing is a form of part-time work in which two (or occasionally more) people share the responsibility for a full-time job. They share the pay and benefits in proportion to the worked hours. Job sharers may work split days, split weeks, or alternate weeks" (ACAS 2010 p. 8).

Employees who do this kind of jobs, sometimes, give quality of life as the main reason why they only want to work part time. After considering the amount of time they spent at work and the amount of time they were able to spend on leisure activities, they opted to work less so they could enjoy life much more. There are also some advantages for the employers, such as skills development, productivity and innovation rise. All the advantages and disadvantages can be found in in MARSHALL (1997).

3.4 Seasonal work

"The seasonal job is a short-term position based on season, or a higher demand of workers for particular parts of the year" (ELLIS-CHRISTENSEN, 2012). "Seasonal work combines a potentially longer-term, relational aspect with short-term, precarious employment and indicates the need to consider not only the linear dimensions of time in assessing contingent work but also its cyclical nature" (AINSWORTH, PRUSS 2009, p. 232).

Even if this kind of work is usually associated to immigration and precarious work, it is an excellent opportunity for people seeking to make some money on their extra time. For example, students can work on their holidays in order to earn something and to begin to enter in the labor market. It is also attractive for both employers and employees because it allows to build relationships and if you've worked well for an employer in a prior season, they're will probably be more inclined to hire you in the future, as they're well aware of your skills and abilities.

3.5 Overtime

A common definition of overtime work is the amount of time someone works more than the normal working hours. For example, more than eight hours in a day. The normal amount of hours can be determined in several ways but the most common ones are by legislation and by agreement between employers and workers (or their representatives). This agreement should be officially ordered and approved by management and is performed by an employee.

There are many laws designed to dissuade or prevent employers from forcing their employees to work excessively long hours. A part from the humanitarian considerations, like preserving the health of workers, it also helps from increasing the overall level of employment in the economy. In Germany, the "State has altogether a minor part in the regulation on working arrangements". The government only defines "minimum conditions which have to be observed compulsory when arranging working time" (CARL, MAIER 2009, p.4). Further information over German work law can be found in that paper.

One common approach is to regulate overtime requiring employers to pay workers at a higher hourly rate for overtime work. Companies may choose to pay workers higher overtime pay even if they are not forced to do so by law, this is very typical when they believe that they face a backward bending supply curve of labor. One of the most common options of overtime work arrangement is Time off in Lieu. According to LONSDALE (2012), Time off in Lieu (TOIL) refers to a type of

work schedule arrangement that allows workers to take time off instead of, or in addition to, receiving overtime pay. So, that the employer, instead of receiving money, will receive time off for each hour worked on certain agreed days.

3.6 Shift-work

"In general, the term 'shift work' is quite vague and includes any organization of working hours that differ from the traditional diurnal work period; sometimes it is a synonymous of irregular or odd working hours" (COSTA 2003, p. 264). The work is organized in shifts. Each of these shifts is bounded in time in order to cover all the hours of operation (generally shifts boundaries are designed so that they don't overlap).

This employment practice is usually designed to keep an organization operating during 24 hours a day, 7 days a week. "Shift work occurs whenever 24 hour coverage is necessary or when a 24 hour day optimizes work output and productivity. Many approaches to shift work exist and each shift work schedule has challenges" (HEATHFIELD 2011).

"Under these definitions, shift-workers include all people working evening shift, night shift, rotating shifts, split shifts, or irregular or on-call schedules both during the week and at weekends" (IWH 2005, p. 1).

In order to have a better idea about what Shift-work is, an example of Shift-work working times will be mentioned: the three shift example. It is used in the warehouse of the "Hospital Universitari Vall d'Hebron" where I worked.

In the case of a 24 hours operation, the workday will usually be divided in three shifts. If the workday of each employee is 8 hours, these are the three different shifts:

- **Morning or first shift.** The morning (or first) shift is the one that starts between 6 a.m. and it finishes at 2 p.m.
- **Afternoon or second Shift.** The Afternoon (or second) shift is the one that starts between 2 p.m. and finishes at 10 p.m. This second shift, occupies the time in during which many people finish work and socialize.
- **Night or third Shift.** The Night (or third) Shift is the one that starts between 10 p.m. and finishes at 6 a.m. This third shift, creates a situation in which the employee must sleep during the day.

This is generally worked over a five-day week, to provide coverage 24/5. The three shift example can be summarized in the following table:

Time	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
06:00 to 14:00	1st Shift	1st Shift	1st Shift	1st Shift	1st Shift	Off	Off
14:00 to 22:00	2nd Shift	2nd Shift	2nd Shift	2nd Shift	2nd Shift	Off	Off
22:00 to 06:00	3rd Shift	3rd Shift	3rd Shift	3rd Shift	3rd Shift	Off	Off

Table 1: Three shift example

Shift work has also two different forms: rotating or fixed shifts.

"Rotating shift schedules mean that employees will regularly switch to a different shift (e.g., days to afternoons to nights)"(OLIVER, CAPSHAW 2012). Here are some examples:
- Traditional rotating schedules: Each crew changes from one shift to another weekly.
- Slow rotation schedules: Rotation is not done automatically every week. The crews may have to change to a new shift every month or even every year.
- Partial rotation schedules: Only a portion of a crew rotates. The others stay on their fixed shift all the time.
- Oscillating schedules: These schedules usually have two shifts that rotate back and forth while the third shift remains fixed.

On a fixed shift schedule, each crew or worker comes into work at the same time every day they are scheduled to work. "Fixed shift schedules mean that employees always work the same shift (e.g., day shift). They may work different days each week, but always on the same shift" (OLIVER, CAPSHAW 2012). Some variations can include a rotation between days off during the week or varying shift lengths.

Acording to CVTIPS (2012), shift-work has some important disadvantages for workers. First of all, workers are forced to work during unsettling schedules different from the normal working hours. This may have a big impact on the workers Work-Life balance (especially for those who work on night shifts). Another point to take into consideration is the fact that the shifts are different and therefore there will be inequality among them. Usually the night workers are gratified with higher remuneration as payoff for these differences.

According to WORKPLACE FLEXIBILY 2010 (p. 3) the workers can usually do some arrangements regarding shifts and breaks.

a) Shift Arrangements.

Workers who are assigned shifts by their employers enter into arrangements with their employers giving them more flexibility regarding the shifts they are assigned. For example, a husband and wife working together enter into an arrangement to ensure their shifts are staggered so that they will have child care coverage for their children.

b) Break Arrangements.

Workers who generally can only take assigned breaks enter into an arrangement with their employers giving them more flexibility over when they take breaks. For example, a worker with diabetes is allowed to set his/her own break schedule in order to ensure an opportunity to eat snacks and meals every three hours.

3.7 Term-time working

"With term-time working an employee works regular hours during school terms but takes time off over school holidays. This kind of work is common in educational environments and is often popular with parents of school-age children" (ALEXANDER 2012). In fact, it is designed primarily for parents with school age children, so they only work when their children are at school. This means, that the parents could be during the summer, Christmas, Easter holidays and half-term breaks (if any) with their children. In summary, it's like any other working time model but with an average of 13 weeks of time off per year.

"Not all employers are keen to set up term-time working schemes. They might be reluctant to find cover for the extra eight or so weeks of absence. But, in some jobs, with seasonal peaks and troughs, an employer might actually prefer you to take up a term-time working scheme. Term-time work is not the same as seasonal work, though. Term-time workers have a permanent contract that continues throughout school holidays and during periods of paid and unpaid leave. This means that your employment rights and advantages are the same as for other employees" (BRENNAN 2012, p.1).

According to LEWIS (2002), term-time jobs have some disadvantages: first of all, a salary reduction. If the employee works less hours, he will earn less money. It also affects the chances of promotion in the company; it is very difficult for a worker to have an important role in an enterprise if he is on holidays a lot of weeks per year.

3.8 Sabbatical

This term comes from Hebrew “sabbat” or “sabbath”, which means literally “ceasing”. In part, it's complicated to classify sabbatical as a working time model because it is a rest from work. But it is also true that sabbaticals are a growing trend and it can't be ignored: “In a 1990 poll conducted for the magazine Special Report, 64 percent of respondents aged 25 to 49 reported that they regularly dreamed about quitting their jobs to go off and do something else for a while” (ROGAK 1994, p. 10).

As seen in COSTER (2010), hundreds of companies, including Boston Consulting Group or eBay, offer paid and unpaid sabbaticals for employees who want to improve their health, recover from job burnout, develop new skills or clarify what they want to do with their lives. Usually, the worker who decides to do so, ceases from two months to a year. In this project, sabbaticals will be modelled as a kind of holiday instead of a working time model.

3.9 Flexible working time models

As an introduction to the flexible working time models, let's have a look at the information furnished by the BMW Group (2002) about working times: only those companies that organize work more efficiently and make more continuous and extensive use of their plant and equipment stay ahead. The flexible organization of operating time and thus working hours represent a key prerequisite in this regard.

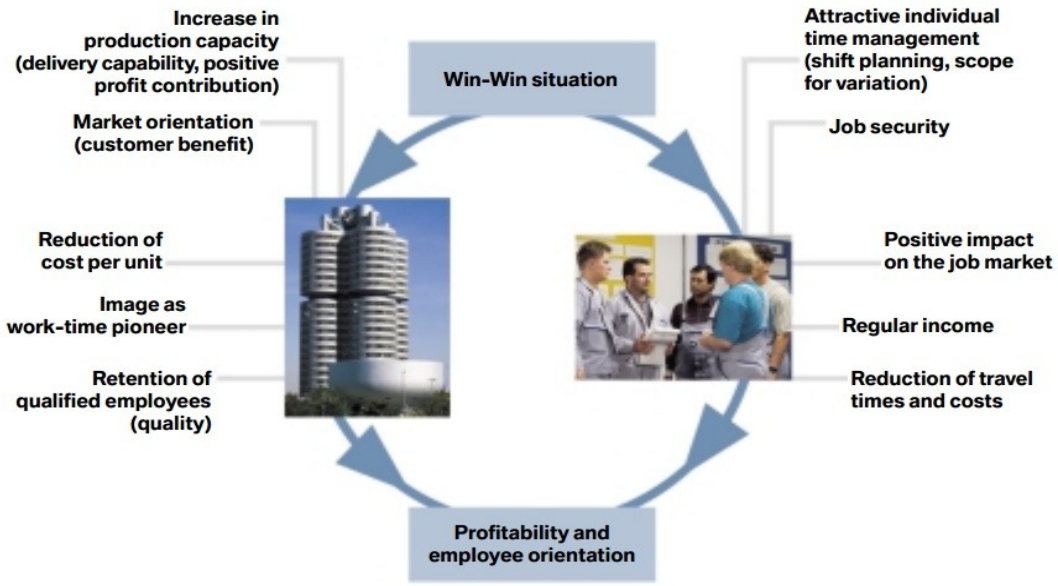


Illustration 7: Flexible work time models (BMW 2002, p. 3)

WORKPLACE FLEXIBILY 2010 (p. 1) mentions that flexible work arrangement include :

1. Flexibility in the scheduling of hours worked, such as alternative work schedules (e.g., flex time and compressed workweeks), and arrangements regarding shift and break schedules;
2. Flexibility in the amount of hours worked, such as part time work and job shares; and
3. Flexibility in the place of work, such as working at home or at a satellite location.

3.9.1 Flextime work

"Flextime, a work schedule that permits flexible starting and quitting times, has gained wide currency as a partial solution to conflicts between work and family life" (CHRISTIANSEN, STAINES 1990 p. 455). Usually flextime, has a "core period" (of approximately 50% of total working period/day) when the employees are expected to be at work. The rest of the time is "flextime", in which the employees can choose when they work (depending on what the employer expects and the amount of work to be done). As seen in LORING, CURTIS (1984, p. 79), this working time model, allows the employee to deviate from the standard work week without supervisor approval.

"In 1997, more than 25 million workers, or 27.6 percent of all full-time wage and salary workers varied their work hours to some degree. The proportion of workers on flexible work schedules—either formal or informal—has more than doubled since 1985, when such data were first collected" (BEERS, 2000, p. 33). This information from the U.S. Department of Labour shows a trend that has increased in the past years and will continue to grow.

The benefits of Flextime can be found in FLEX PLANNER (2012) and HEATHFIELD (2012). The following benefits for the business can be listed:

- Greater staff morale and job satisfaction. Most employers offering flexitime working report improvements in recruitment, absenteeism and productivity
- Reduces stress and fatigue and unfocussed employees
- Increases employee satisfaction and production
- Measures employee's attendance – you only pay or time in attendance (delayed arrival caused by traffic congestion, delayed trains, doctor visits etc. are at employees expense)

And for the employees:

- Increased opportunity to fit other commitments and activities in with work, and make better use of their free time.

- More in control of their workloads, and manage a better balance between life and work. Allows you to schedule your travel; time to avoid congestion
- Allows you bank time to be used for leisure / personal activities
- Increased feeling of personal control over schedule and work environment.

There are many different models of flexible working time schedules. We are going to see some of them but all can be checked at OPM (2012).

3.9.1.1 Gliding Schedule

- Basic Work Requirement: A full-time employee must work 8 hours a day, 40 hours a week, and 80 hours a biweekly pay period. The agency head determines the number of hours a part-time employee must work in a day, in a week, or in a biweekly pay period.
- Tour of Duty: Employers establish flexible and core hours. Gliding Schedules (means a type of flexible work schedule in which a full-time employee has a basic work requirement of 8 hours in each day and 40 hours in each week, may select a starting and stopping time each day, and may change starting and stopping times daily within the established flexible hours) provide for flexible time brands at the start and end of the workday and may also allow for flexible hours at midday (during the lunch break). Employees must work during core hours.
- Core hours: An employee must account for missed core hours (means the time periods during the period that are within the tour of duty during which an employee is required by the agency to be present for work) with leave, credit hours, or compensatory time off.
- Overtime work: Overtime work is work in excess of 8 hours in a day or 40 hours in a workweek, ordered in advance by management.
- Flexibility: Employees may vary arrival and departure times on a daily basis during the established flexible hours.
- Schedules: Check the picture below.

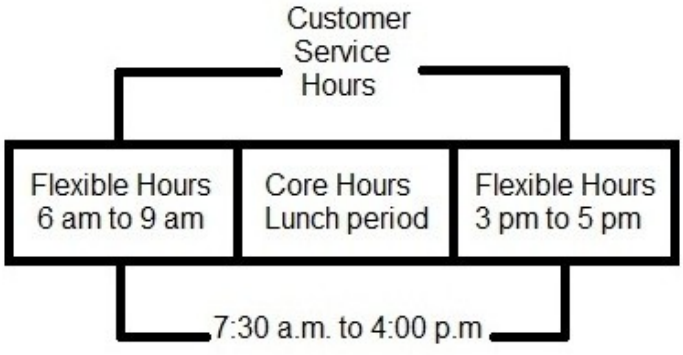


Illustration 8: Example of Gliding Schedule (OPM 2012)

3.9.1.2 Variable day Schedule

- Basic Work Requirement: A full-time employee must work 40 hours a week. The agency head determines the number of hours a part-time employee must work in a week.
- Tour of duty: Agencies establish flexible and core hours. Gliding Schedules provide for flexible time brands at the start and end of the workday and may also allow for flexible hours at midday (during the lunch break). Employees must work during core hours.
- Core hours: An employee must account for missed core hours (means the time periods during the workday, workweek, or pay period that are within the tour of duty during which an employee covered by a flexible work schedule is required by the agency to be present for work) (if permitted) with leave, credit hours, or compensatory time off.
- Overtime work: Overtime work is work in excess of 8 hours in a day or 40 hours in a workweek, ordered in advance by management.
- Flexibility: Employees may vary arrival and departure times on a daily basis during the established flexible hours. An employee may also vary the length of the workday. An agency may limit the number of hours an employee may work on a daily basis.
- Schedule: Check the following picture.

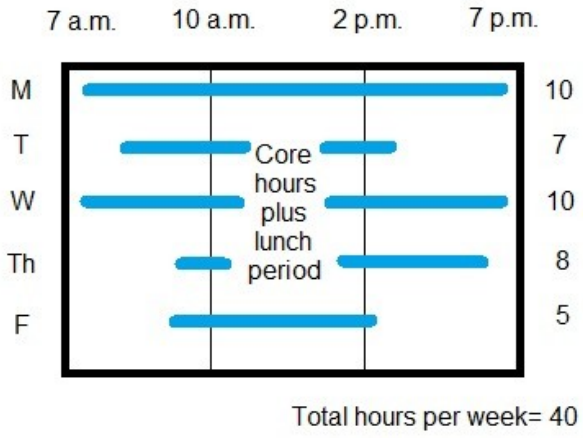


Illustration 9: Example of variable day schedule (OPM 2012)

3.9.1.3 Variable week Schedule

- Basic Work Requirement: A full-time employee must work 80 hours in a biweekly pay period. The agency head determines the number of hours a part-time employee must work in a biweekly pay period.
- Tour of duty: Agencies establish flexible and core hours. Gliding Schedules provide for flexible time brands at the start and end of the workday and may also allow for flexible hours at midday (during the lunch break). Employees must work during core hours.
- Core hours: An employee must account for missed core hours (if permitted) with leave, credit hours, or compensatory time off.
- Overtime work: Overtime work is work in excess of 8 hours in a day or 40 hours in a workweek, ordered in advance by management.
- Flexibility: Employees may vary arrival and departure times on a daily basis during the established flexible hours. An employee may also vary the length of the workweek.
- Schedule: Check the picture on the next page.

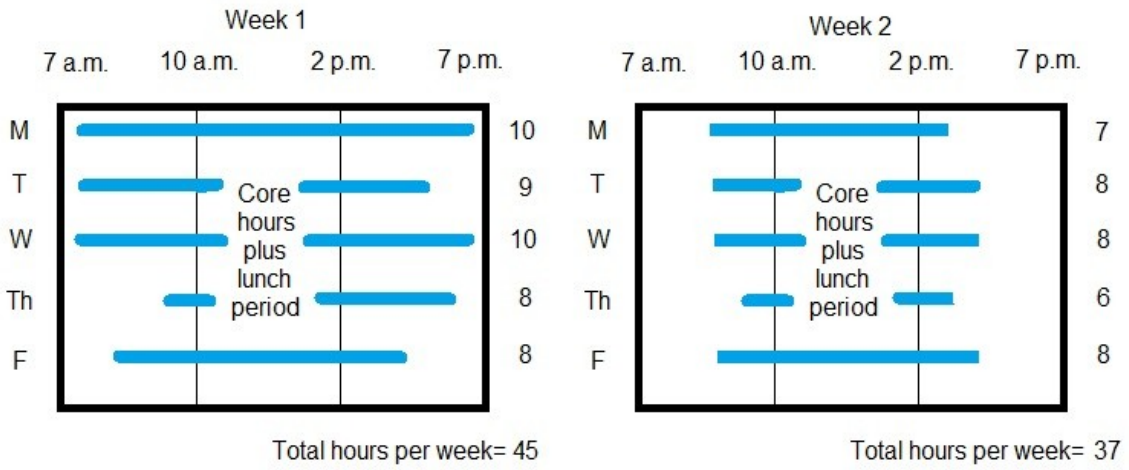


Illustration 10: Example of variable week schedule (OPM 2012)

So far, in the schedules presented, the employer can plan the whole week, fix some core hours and give some flexibility to the employees. In flexible work schedules without core hours, the working time of the employee is not divided in core and flexible hours, so the employee has full flexibility.

3.9.2 Compressed work week

"A compressed work week is an arrangement whereby employees work longer shifts in exchange for a reduction in the number of working days in their work cycle (i.e. on a weekly or biweekly basis)" (SIMCHA 1984, pp.116).

“Compressed working weeks involve the relocation of time worked into fewer and longer blocks during the week. This does not necessarily involve a reduction in total hours worked or any extension in individual choice over which hours are worked” (ACAS, 2010, p. 9). Working a longer number of hours over four days, instead of over five days is a common arrangement. In a certain way, employees can build up additional hours which they take as a day or half-day away from work.

Summarizing the information by ACAS (2010) and SIMCHA (1984), the following advantages for employees can be listed:

- Possibility of additional days off work (e.g. longer weekends allowing “mini-vacations”) and reduced commuting time, whereas employers can extend their daily operating hours with less need to resort to overtime.
- Particularly useful for employees who wish to reduce the number of days per week spent at work, but who cannot financially afford to decrease their working hours.
- Job Satisfaction
- A compressed work week can lead to lower absenteeism rates because it gives employees more time to go to appointments and fulfill other obligations.

According to KATEPOO (2012), here are some disadvantages of the compressed work week:

- An ongoing schedule of ten-hour or nine-hour days, while it may be the norm for some professionals already, can be physically and mentally draining.
- Not only is the workweek squeezed into a shorter time frame, but all the after-work activities must also be wedged into the remaining hours of each work day.
- Chronic fatigue caused by current work-family conflict time pressures may or may not be off-set by the regular day off.

Disadvantages for employers: (See all in HUMPHREY 2012)

- Longer daily hours can cause greater physical and mental exertion, requiring sufficient stamina and energy. This can also raise concerns with respect to health and safety in the workplace, since risks are normally higher when employees are fatigued. These arrangements may therefore not always be appropriate in occupations requiring a high level of precision and sustained attention, especially in the handling of hazardous substances or equipment.

- Some workers on extended work days may pace themselves differently than workers on traditional shifts. Other employees may slow down at the end of a day as fatigue sets in. Most workers on four-day weeks need more breaks or longer breaks. These factors may mean less work is done in a week when compared to a traditional shift.
- Although employees may work four-day weeks, most businesses cannot function on the same schedule.
- When everyone works flexible schedules, finding a time when everyone is available becomes more difficult.
- In longer shifts more errors of the employees.

3.9.3 Annualized hours

“Flexible working time arrangements can also be provided in the form of "annualized" hours. These essentially allow employees to choose, within certain boundaries, their days and hours of work, with the proviso that they work a specified number of hours in a year. This can also be calculated over a shorter averaging period, be it on a monthly, biweekly, or other basis. Such arrangements combine elements of flex time and compressed workweeks and can have the added advantage of reducing recourse to overtime” (SAINT-CYR, 2007).

“This pattern originally developed in industries with a seasonal work flow, such as manufacturing, but has extended into retailing, financial services, and health and emergency services” (ALEXANDER, 2012). Annualized hours contracts are normally (but not always) associated with shift work. According to ACAS (2010, p. 11), the annual hours an employee is contracted to work are normally split into:

• set shifts which cover the majority of the year

• inallocated shifts which the employee can be asked to work at short notice (these remaining hours are kept in reserve so that workers can be called in at short notice as required).

“Employers may also benefit from such arrangements if they can be used to meet seasonal variations and peak hours. Moreover, output and product quality can be improved if employees work during their most productive periods of the day” (SAINT-CYR, 2007). Other benefits for the employers are the improvement of competitiveness and productivity, costs reduction (among others because no overtime hours are paid).

On the employees side, annualized working hours guarantees a stable income regardless of the weekly worked hours and a greater flexibility. Some disadvantages are the irregular and sometimes

unpredictable working hours. The whole set of disadvantages and advantages for both employees and employers can be found at ILO (2004 p. 3).

3.9.4 Hot-desking

"Hot-desking originates from the definition of being the temporary physical occupant of a work station or surface by a particular employee. The term hot-desking is thought to be derived from the naval practice, called hot racking, where sailors on different shifts share bunks. Originating as a trend in the late 1980s-early 1990s, hot-desking involves one desk shared between several people who use the desk at different times" (NUBEY 2009, p. 272).

Hot-desking presents many advantages for both employee and employer. The primary motivation for hot-desking is cost reduction though space savings. Also forces workers to be better organized as they don't have a concrete place for making castles of documents for example or leave unattended stuff. Usually, the work surface is just a terminal link and the employee can work from anywhere (from home or "on the move").

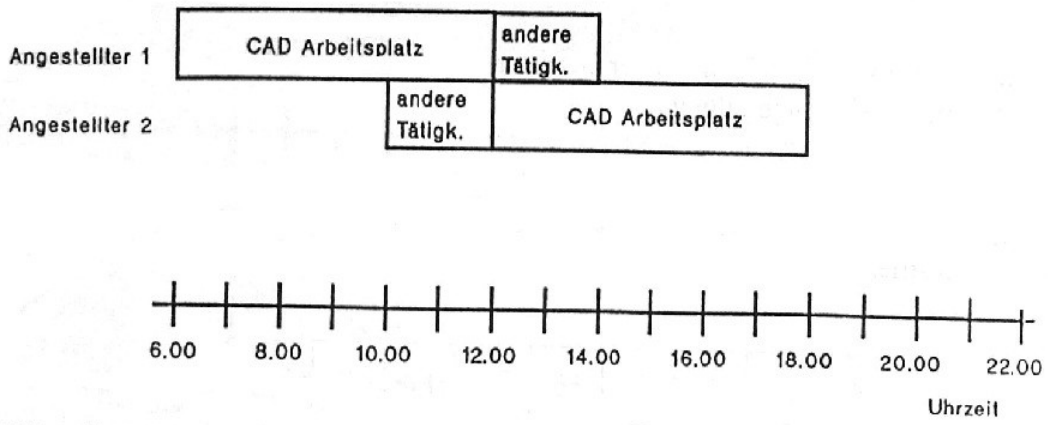


Abb. 4.2-5: Gestaffelte Arbeitszeiten für eine Ausdehnung der täglichen Nutzungsdauer eines CAD-Arbeitsplatzes auf 12 Stunden (D in Abb. 4.2-1)

Illustration 11: Staggered working hours for an extension of the daily life of a CAD workstation to 12 hours (BECKER, EISSSING, EULER 1993, p. 480)

The picture above can be seen as an example where two workers ("Angestellter 1" and "Angestellter 2") share a common work station ("CAD Arbeitsplatz). The main idea of the diagram is to see that both employees can work 8 hours per day without overlapping.

3.9.5 Phased retirement

"There is no single, recognized definition of phased retirement. The term may be described as a broad range of employment arrangements, formal and informal, that allow an employee who is approaching normal retirement, to continue working, usually with a reduced workload, in transition from full-time work to full-time retirement" (BRAINARD 2002 p. 1).

Phased retirement takes many forms, including part-time, seasonal, or temporary work, an extended leave of absence or a deferred retirement option plan" (BRAINARD 2002 p. 1). We decided to classify it as flexible working time model because the employee can also adopt tele-work or flexible work time, for example.

According to SHIFTWORK Ltd. (2012), the retiree can act as a mentor to other younger workers of the staff. It results in a smoother transition of responsibilities and a better dissemination of knowledge and experience. Moreover, for the employer this means that he can retain trained and qualified personnel and reduces costs at the same time through lower salary.

3.9.6 V-time (Voluntary Reduced Time) working

"V-time working involves a voluntary reduction in hours for a fixed period with the guarantee of a return to normal hours once the period ends. V-time working may be initiated by the employee or the employer, and is normally agreed for specific purposes, e.g. undertaking a course of study or caring for a family member" (ALEXANDER 2012). Of course the reduction of working hours comes along with a salary reduction.

"Organizations can benefit from this arrangement as they can make savings in downturns of the business cycle. Employees appreciate the option even if they choose not to take it up but remain on full-time hours" (SHIFTWORK Ltd. , 2012). It brings a new level of flexibility to the security of a full-time job. This suits people with a wide range of needs: those who want to spend more time with their children or simply dedicate more time to themselves or to develop their skills in new directions.

CARTER (2012, p. 6) says that there are many ways in which working time can be reduced and directors may agree to any request which is considered reasonable. Possible options are as follows:

- Reduce to a four day week

- Reduce daily hours
- Reduce hours for specific parts of the year
- Work in term-time only

These options can be for a specified period only or a long term arrangement subject to satisfactory reviews. The agreed work pattern will normally be maintained consistently and will only be varied in exceptional circumstances following full consultation.

3.9.7 Tele-work

"Teleworking, also known as telecommuting or flexiplace, is a 21st century and beyond business solution, incorporating the newest technologies to enable productive work away from the standard office atmosphere. With technology developing at immeasurable speeds, tele-work offers a fresh, new alternative for many." (TELEWORK.ORG 2012, p. 1).

"Tele-work is basically a technological system that allows workers to both enjoy the comforts of home while seamless to the organization of customer, conducting their normal jobs" (TELEWORK.ORG 2012, p. 1).

"Tele-work can, among other things, help generate extra productivity; save on expenses; and improve employee retention. Allowing your employees to tele-work not only strengthens the trust of current employees, but also attracts new employees looking for flexibility in the workplace" (TELEWORK.ORG 2012, p. 1).

Most of the information about tele-work comes from governments and most of them are focusing their efforts in this way. For example in the U.S.A., "Increasing the strategic use of telework is a high priority for President Obama and OPM, and we appreciate the strong support of the Congress, as evidenced by passage of the Tele-work Enhancement Act of2010 (the Act)" (OPM 2010 p. 2).

Tele-work can be divided in three different types:

- Work from home (using telecommunications if necessary)
- Work from a satellite location (the company sends the employee away to a satellite location of the firm in another city, for example).
- And work from different locations (the company sends the employee to different locations depending on the requirements of the firm).

After the realization of the inventory of working time models, the next chapter will show a modeling of those working time models. This model concept will help the reader of this diploma thesis to comprehend the different working time models that will be taken into account.

4. Working time models modeling

In this chapter a little introduction to the modeling language UML (Unified Modeling Language) will be done. There will be also an explanation about how the woking time models will be modeled and why. At the end, the software requierements will be presented.

First of all, the tools used for modeling will be presented. In fact, only a single tool will be used for modeling: the Unified Modeling Language (UML) since it's a standard in the systems engineering.

4.1 Introduction to UML

According to FOWLER (2004, p. 19), the UML is a family of graphical notations, supported by a unique Meta-model. The notations help in describing and developing software systems, especially when these systems are object-oriented. The Unified Modeling Language is a general-purpose visual modeling language that is used to specify, visualize, construct, and document the artifacts of a software system" (RUMBAUGH, JACOBSON 1998, p. 3).

"UML was developed in an effort to simplify and consolidate the large number of object-oriented development methods that had emerged" (RUMBAUGH, JACOBSON 1998, p. 4). "The language has gained significant industry support from various organizations via the UML Partners Consortium and has been submitted to and approved by the Object Management Group (OMG) as a standard" (ALHIR 1998, p. ix).

Our main goal is not to make a complete tutorial about UML but just explaining what tools and why they are useful. From here, I am going to focus on the model and the way I developed it.

4.2 First conception

"Creating models is highly creative work. There is no final solution, no correct answer that is checked at the end of the work. The models designers, through iterative work, assure that their models achieve the goals and the requirements of the project under construction. But a model is not final; it's typically changed and updated throughout a project to reflect new insights and experiences of the designers." (ERIKSSON, PENKER 1998, p.1). Therefore, I will try to explain which steps I take to reach the final model.

“Generalization/specialization allows classes to share the same code. This reduces the code size and provides for more maintainable software” (LEE, TEPFENHART 1997, p. 12). Generalization is one of the most used tools in object-oriented models and after a brainstorming, I tried to use it in our model as in the picture below:

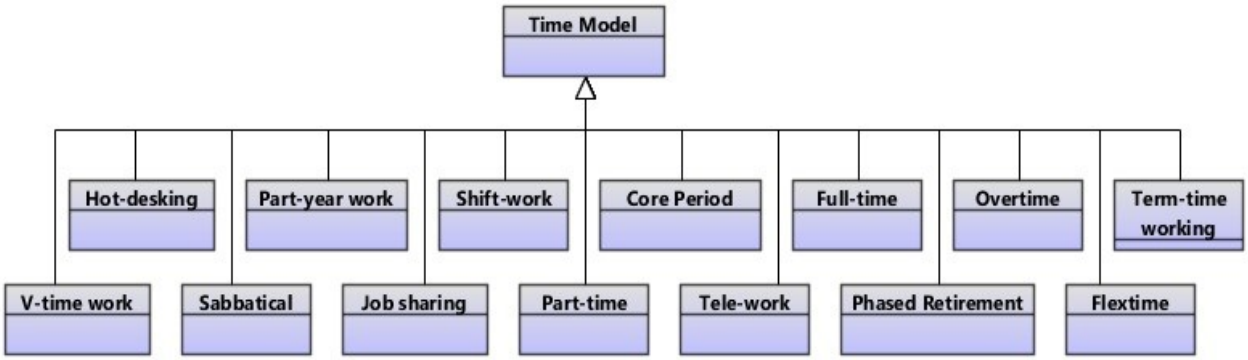


Illustration 12: First idea for the time models UML model (only with classes names)

The UML looks like a big tree with the class “Time Model” as root node. All the other classes inherit from it. In other words, all the concrete time models are children from “Time Model”. But this idea became more and more complicated to take into account because the model became confused and rambling and each independent class should have its own implementation (with its own attributes and functions). Furthermore, some time models do not need an own class because they don't need special attributes. For example, “Tele-work” or “Hot-desking” can be a normal Time Model with the difference that the location attribute could change.

4.3 Tune up

For all the reasons explained above, a single Time Model class who contains all the time models attributes such as “name”, “start” or “location” was created. The name will be the differentiation factor for all the time models. The “Core Period” time model will be the only specialization of the model because it has a unique attribute: the core period week. This attribute permit to model the hours when the worker has to be at the work (see Chapter 3.2 for more information).

In fact the “Core Period” will allow us to model all the flexible time models like “Annualized working hours” thanks to the attributes (“hoursPer” defines the number of “hours” that the worker should work per year, month or week). A “Compressed work week” will be modeled as a normal “Time Model” but with a 4-days-“Week Shift”. And finally, “Phased Retirement” can be modeled

as a combination of any other time model depending on the agreements between worker and employer.

With all this, our new time models modeling looks like follows:

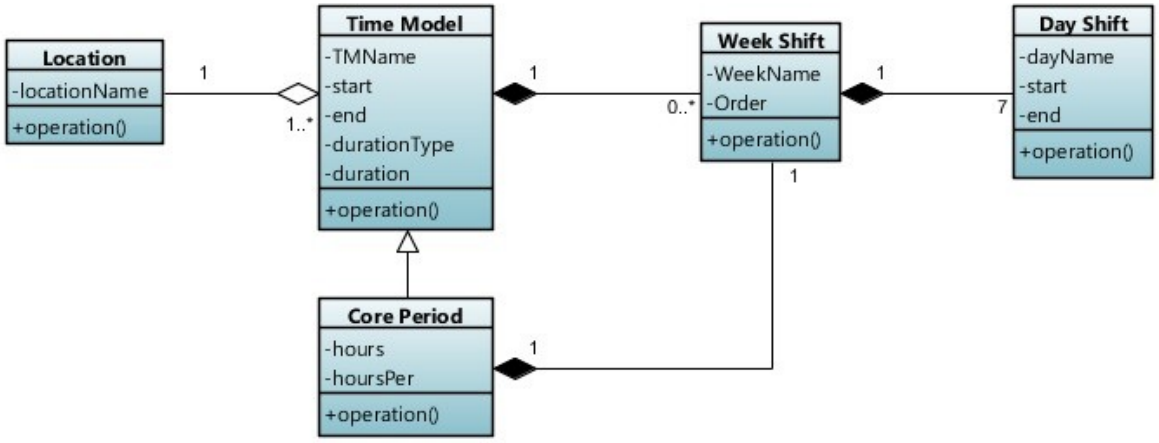


Illustration 13: Time model UML 2nd version (only with attributes names)

Some association relations between the classes were added. For example, each “Time Model” has a “Location” associated. This relation (with the white diamond) is called Aggregation and refers to a “part-whole” or “Is-part-of” link. More information about these relations can be found in FORBRIG (2007, p.24). This only means that a “Location” is a part of the “Time Model”. There is also some compositions (with the black diamonds on the illustration) : “a composition is a stronger relation than a Aggregation” (FORBRIG 2007, p.24). For example, a “Week Shift” means nothing if not associated to a “Time Model”. Therefore, if the “Week Shift” is not associated, it won't simply exist. The same happens between “Week Shift” and “Day Shift”. On the other side, a “Location” can still exists event if not associated to a concrete “Time Model”.

In regard to the concrete modeling, the “Full-time” and “Part-time” time models can be thought as a “Time Model” with only one “Week Shift” (with the concrete starting and ending hours per day). As “Job sharing” time model is also a form of part-time working, it can also be modeled. “Term-time working” can be modeled as any other time model but with more vacation days. “Part-year (or seasonal) Work” can also be modeled as a “Time Model” with a concrete ending date. “Shift-work” will be represented as a “Time Model” with some different “Week Shift”.

Until here, everything sounds interesting but I haven't yet talked about the worker and a working time modeling without workers won't make any sense. How can the worker be included in the project?

4.4 Adding the workers

the workers should be introduced in the model because, for example, “Overtime” is inherent to a concrete worker. A working time model called “Overtime” is meaningless without an employee who works during a concrete amount of time (in addition to normal working hours). The UML model with workers look like this:

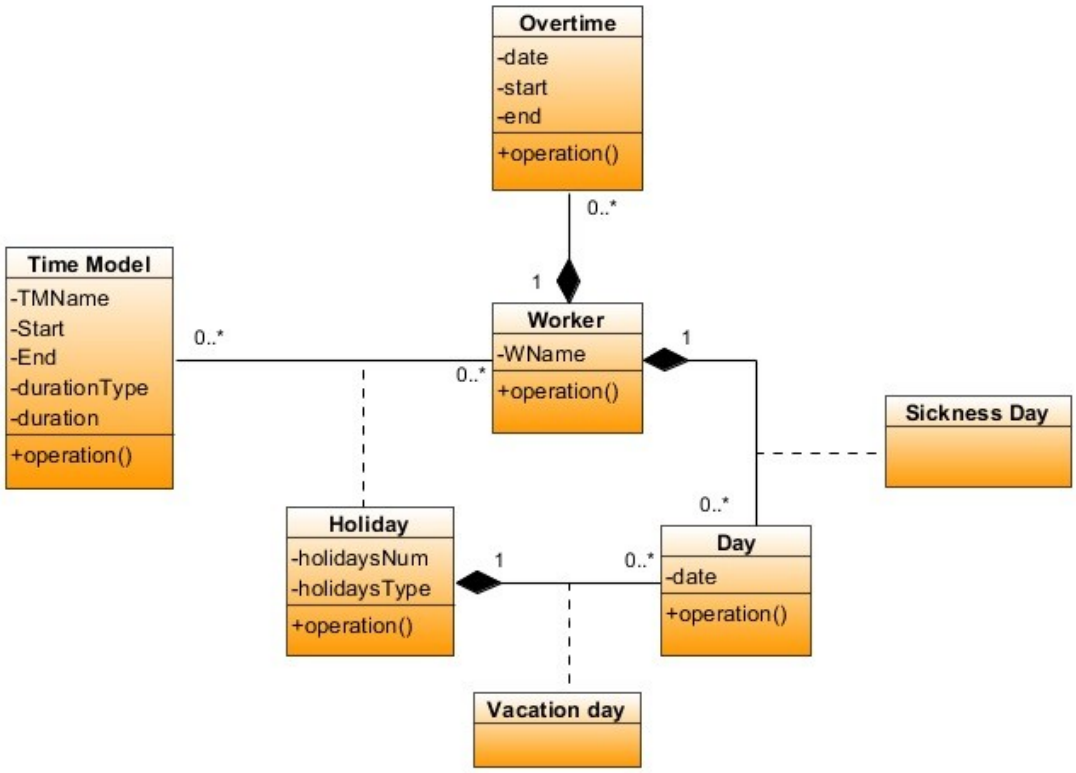


Illustration 14: UML model with the "Worker" associations (only with attributes names)

The picture above shows the UML model of the worker and his associations. We can see that there is an association class called “Sickness Day” which represents when a worker is on a sick leave. “Association classes are associations with class properties or classes with association properties” (ALHIR 1998, p. 149). For example, if the instance of a “Worker” stops existing, then all the associated “Sickness Day” will also disappear.

Moreover, other associations were added to the model to adapt the model to the specifications including “Holiday” and “Overtime”. In the concrete case of “Holiday”, if the worker decides to take a sabbatical period, then the “holidaysType” attribute will be updated to “Sabbatical”.

If the second and the third pictures are joined uo, the final model appear.

4.5 Final model

The picture below shows a final model simplified (without attributes types and operations). A completed model can be found in the Appendix.

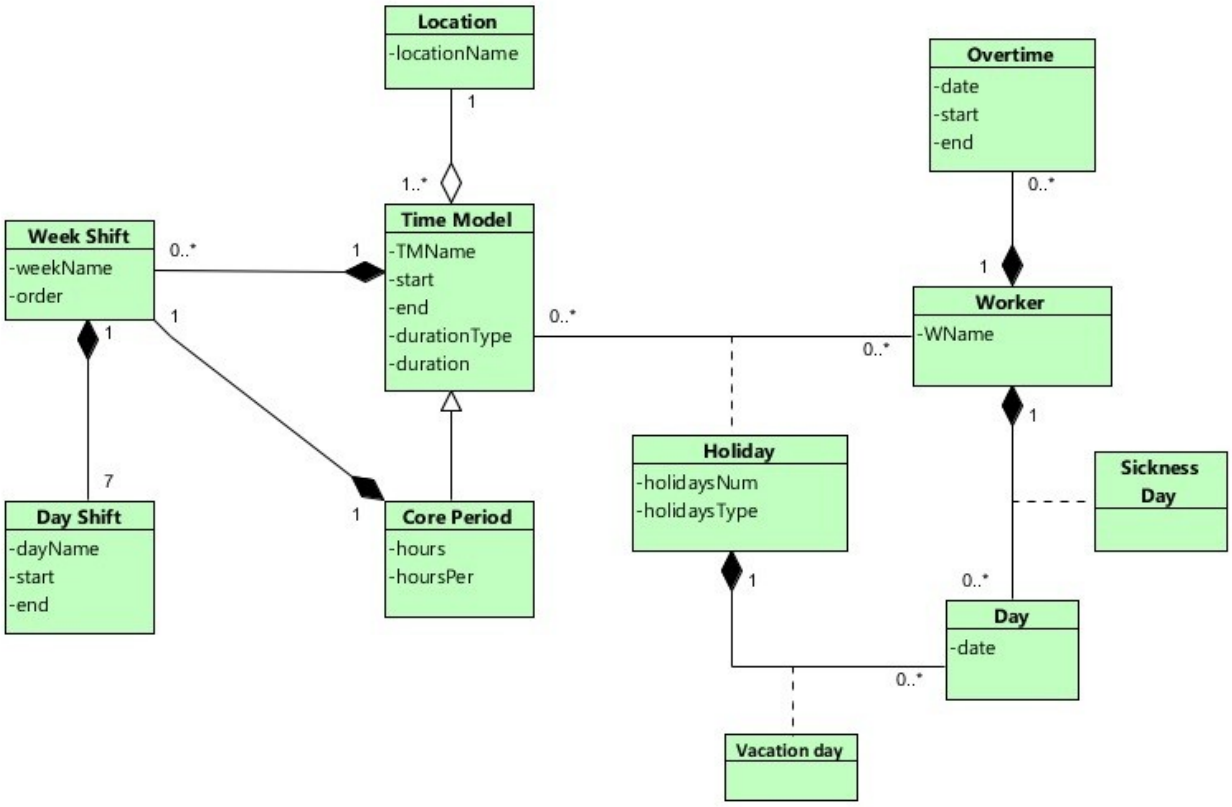


Illustration 15: Final model (only with attributes names)

4.6 Software requirements

In software engineering, the software requirements specification (SRS) “is a specification for a particular software product, program, or set of programs that performs certain functions in a specific environment” (IEEE Std 830 1998, p. 3). It should be a complete description of the behavior of a system to be developed. “In addition to the functional requirements, the SRS contains nonfunctional

requirements. These include performance goals and descriptions of quality attributes" (WIEGERS 2003, p. 11).

4.6.1 Functional requirements

"Functional requirements should define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs" (IEEE Std 830 1998, p. 16). In other words, they may be calculations, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Generally, functional requirements (a description of the software functions) are expressed in the form "system must do < requirement > ".

In our project, the system must:

- Create a new time model with the attributes introduced by the user.
- Create a new worker with the attributes introduced by the user.
- Modify or delete a time model according to the user.
- Modify or delete a worker according to the user.
- Display a view of the list of the time models and the users
- Display a daily calendar of the workers activities
- Display a roster of the workers

Functional requirements may also include a set of use cases that describe interactions between the users and the software. Use cases describes the behavioral requirements in each concrete action. As our functional requirements are simple (only in-/output methods), I will skip this step.

4.6.2 Non-functional requirements

Non-functional requirements (also known as quality requirements) impose constraints on the design or implementation (such as performance requirements, security, or reliability). They are expressed in the form "system shall be < requirement > ". "Quality attributes augment the description of the product's functionality by describing the product's characteristics in various dimensions that are important either to the users or to developers" (WIEGERS 2003, p. 11).

According to WIEGERS (2003, p. 217), HOFFMANN (2008, p. 6-10) and COSTAL COSTA, SANCHO SAMSÓ and TENIENTE LÓPEZ (2003, p. 22-23), our system shall have as characteristics:

- Availability (the proportion of time a system is in a functioning condition);
- Efficiency (resource consumption for given load);
- Flexibility (capacity to adapt when external changes occur);
- Integrity (assurance that under all conditions a system is based on the logical correctness and reliability);
- Interoperability (property of a system to work with other systems, present or future, without problems);
- Reliability (the software executes without failure or for a specific period of time);
- Robustness (the degree to which a system continues to functions properly when confronted with invalid outputs, defects, or unexpected operating conditions);
- Usability (how easy it is for user to use or learn to use the software);
- Maintainability (the ease with which maintenance of a system can be performed);
- Portability (the usability of the same software in different environments);
- Re-usability (how easy it is for programmers to extend the software);
- Testability (the capability of a system to be tested).

There are many non-functional requirements and, in an ideal universe, every system would exhibit the maximum value for all them. But to be realistic, I should focus on some requirements and the most important should be: reliability, robustness, usability and re-usability.

Why did I choose those four requirements? reliability and robustness are essential characteristics of any software system and are way more important than the others. Availability, flexibility, interoperability and integrity are central for any on-line application because Internet is a continuously changing world full of threats. Efficiency is also very important, but more for big systems which require lots of resources than little programs like ours. Maintainability and testability are also important factors (specially for programmers) but as said, our program is kind of simple and finding and solving errors should not be a big issue. The portability of the program is assured by the Qt and C++ code. The last two, usability and re-usability, are the other important factors because our project should be further developed in the future.

5. Implementation and deployment

This chapter will contain the explation of how the implementation of the design was done. Implementation includes all the processes involved in getting the new software operating properly in its environment. In an information technology context, that comprises programming, running, testing, and applying changes (if necessary). The word deployment refers to installation, configuration and utilization of the software. Sometimes both words are used to mean the same thing.

We will divide this chapter in 3 parts. In the first one, the system, the programming decisions and the structure of the program itself will be presented. In the second one, I will analyze the database. Then, in the last part, I will talk about the interface. This chapter mans not to be a complete guide or tutorial over the different technologies or applications used in this project. It only presents some interesting features for the development and understanding of the project.

5.1 System

The system cannot be more simple in this project. As the main goal is to model the working time models, the system has to worry only to get the right data from the input displays and transfer it to the database and vice versa. It can be summarized in the following picture:

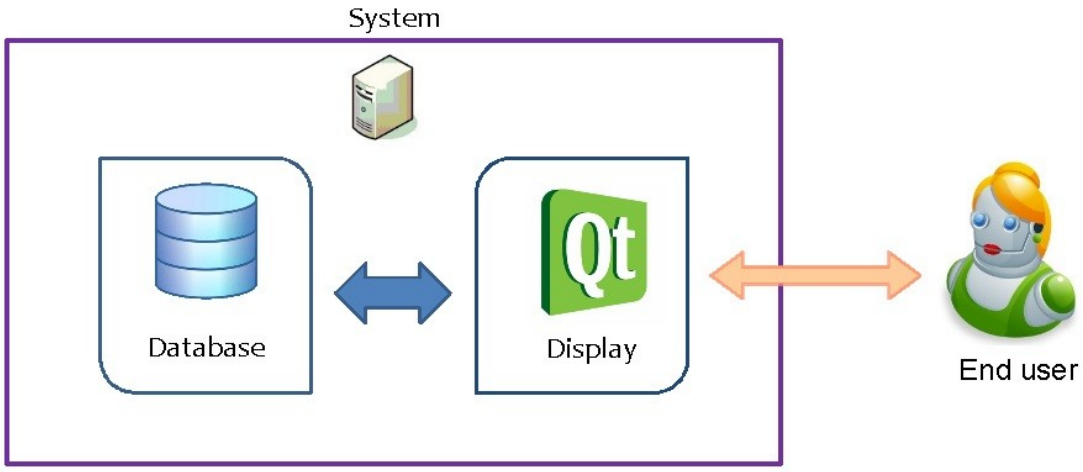


Illustration 16: Schema with the relations between user, system, database and display

5.1.1 Software requirements

The system has to make sure that the data entered by the user and the data showed by the display are correct. Usually, this kind of task is distributed between the display and the system. For example, the display is in charge that the user selects a valid real date for a sick leave (54/13/2012 is not valid); the system is in charge that the date introduced by the user is not a worker's holiday date. This can be seen as a way to match the reliability requirement.

In fact, I tried to match all the software requirements refered in the previous chapter. In terms of reliability and robustness, I tried our best to write a stable program. For example, an important point for robustness is exception management: a proper exception handling cannot allow exceptions throw out to the user level. The exceptions are handled as soon as possible. Considering that this program is offline and relatively small, problems referring to transactions, services, multithreading, logging or external attacks can be put asside.

In regard to the other two important requirements, our program is very usable as the results of our survey show (check chapter 6). I also tried to program focusing on the code modularity and implementing the model following the indications of DEITEL and DEITEL (2008). This way, the re-usability of the software is assured

Qt and C++ code guarantee the portability of the program (check previous chapter for more information). Those both technologies will now be presented.

5.1.2 C++ basics

OUALLINE (2003, p. 3) wrote that C++ is a high-level programming language that allows a software engineer to efficiently communicate with a computer. Since its creation in 1980, it has been used overall in the computer industry (firmware for microcontrollers, operative systems, graphics programming, …).

According to ISERNHAGEN and HARTMUT (2004, p. 4), the most known and important benefits are that C++ is:

- A flexible and adaptable language.
- Memory and run-time efficient.
- Has a high bandwidth (assembler-level to high level languages).
- Standardized and very widespread.

In addition to these advantages, C++ has also a very important characteristic for our project: Qt is a framework originally designed for C++. That's one of the main reasons why Qt was chosen for our display.

To finish, let's have a look to the following schema describing the typical C++ environment:

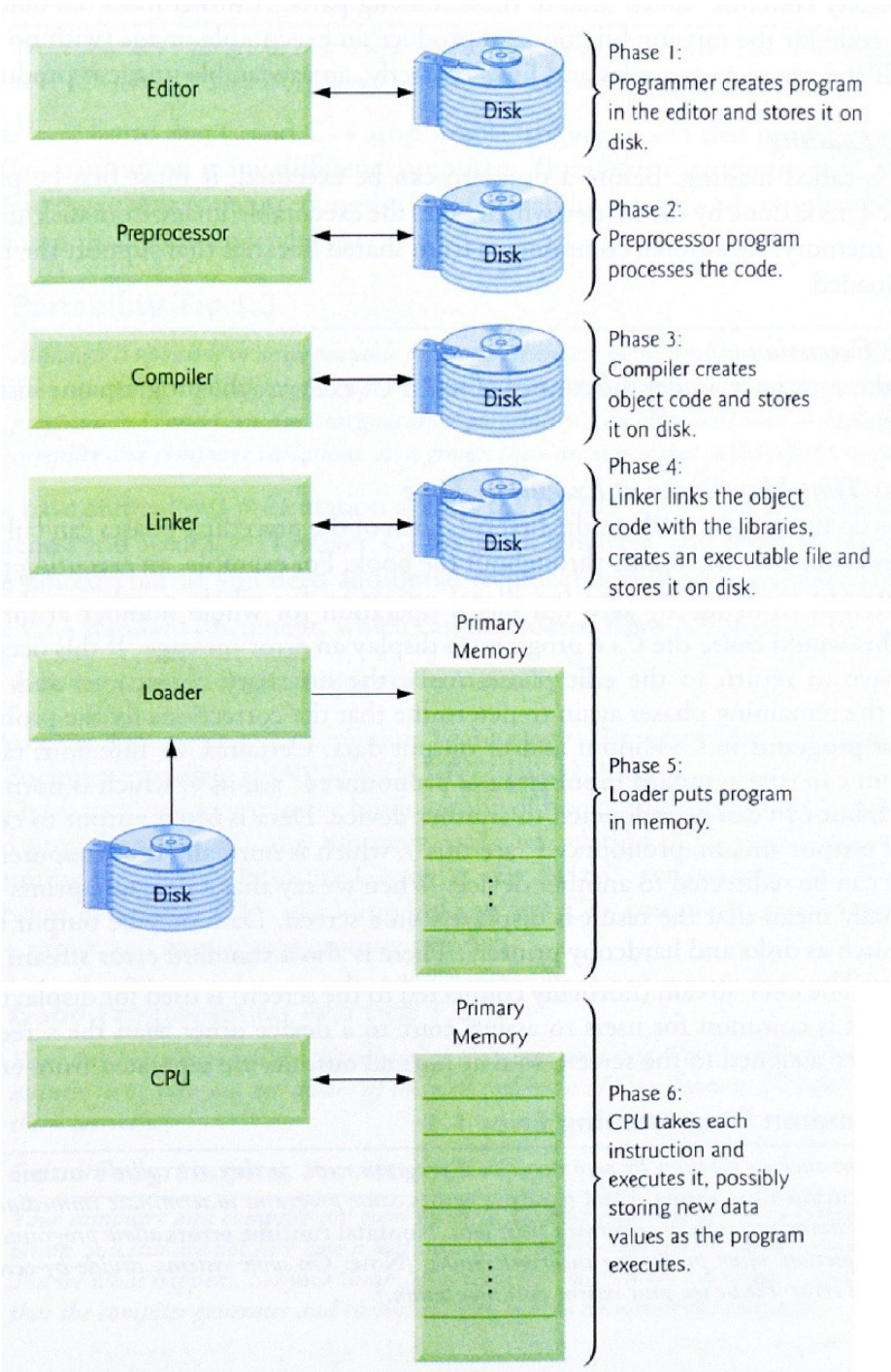


Illustration 17: Typical C++ environment, first part (DEITEL and DEITEL 2008, p.15)

The picture above shows the different steps in creating a C++ application and also the different steps in executing a C++ application.

5.1.3 Implementation decisions on the model

The “Location” class will not be implemented being that it only has one string attribute. We will add another attribute to the “Time Model” class called “loc” with type QString. The question that this decision raises is: Why designing an independent class if it wasn't going to be implemented? It is because of future prospects. The “Location” class should be thought as a class that can contain a lot more that just a location. In future extensions, this class can also hold the address, the location's description, the location's characteristics or even morph into a “Workstation”. Check chapter 7 for more information about that.

In the same direction, the class “Day”, “Vacation day” and “Sickness day” won't be implemented. For the first, as it already exists a class called “QDate” on the standard Qt libraries, it won't make sense to implement our own class. This class fulfills all our project requirements and fit perfectly. The “QDate” class specifications can be found on the Qt documentation on Internet or in handbooks. “Vacation day” and “Sickness day” are association classes and it's interesting to “use the associations described in the class diagram to declare references (or pointers, where appropiate) to other objects” (DEITEL and DEITEL, 2008 p. 518). Therefore, as the “Day” class won't be implemented, the “Vacation day” class and the “Sickness day” class will be added (like attributes) both in the “Worker”.

Our last implementation decision is about the “Core Period” class. This class is supposed to act as an expansion of the “Time Model” class because it contains more attributes and functions. During the implementation, the possibility to manually switch from one class to another appeared. In other words, the user can decide what kind of time model is he defining at any moment. From a programming point of view, to switch from two different classes continuously can be very inefficient (switching means that the system deletes the old class and creates a new one with the same attributes). Therefore I decided that both classes will be merged in one. This class will contain all the attributes of both classes and an activation boolean. If the user decides to change the time models and add a core week only this element will change.

5.1.4 Implementation classes

We are going to analyze a little bit the classes of our program. We will split them in two: the normal C++ classes and the Qt's C++ classes. Let's have a look to the files of the first ones:

```
worker.h          worker.cpp
dayshift.h        dayshift.cpp
overtime.h        overtime.cpp
timemodel.h       timemodel.cpp
weekshift.h       weekshift.cpp
main.cpp
```

Those classes are the direct C++ implementation of the UML classes of our model. All these classes contain the same functions: constructors, “getters” and “setters”. Constructors are the methods which allows us to create an element of the class, “getters” are the methods which allow us to get and access to attributes of the element and “setters” are the methods which allow us to set and modify the attributes of a concrete element. Finally, let's explain a little bit the main of the program: it is very simple, the only thing it does is start the main window of our display. Let's have a look at the Qt's C++ classes of the project listed below:

```
mainwindow.h            mainwindow.cpp          mainwindow.ui
addtimemodeldialog.h    addtimemodeldialog.cpp  addtimemodeldialog.ui
addweekshiftdialog.h    addweekshiftdialog.cpp  addweekshiftdialog.ui
newdayshiftdialog.h     newdayshiftdialog.cpp   newdayshiftdialog.ui
```

Each class has a header, a source and a user interface file. The user interface files contain the forms description of each window or dialog of our program. In addition to these, there is also a “project” file (with the extension “.pro”) which contains all the information required by qmake to build your application, library, or plugin. The command line qmake provides a project-oriented system for managing the build process for applications, libraries, and other components. Fortunately, all these steps are automated in the program Qt creator which will be explained in the next pages.

5.2 In/output interface (display)

5.2.1 Qt

The whole display will be programmed in Qt. As BLANCHETTE and SUMMERFIELD (2009, p. 17) wrote, Qt is it's extensive C++ framework for developing cross-platform GUI applications using the approach “write once, compile anywhere”. Qt is a very powerful tool for programmers to program applications for Windows, Mac, Linux, Solaris and many other operative systems versions based on Unix.

Why Qt? The answer to this question can also be found in the book from BLANCHETTE and SUMMERFIELD (2009, p.13). More precisely, at the preface written by Matthias Ettrich for the

German edition. Some answers to the question “Why programmers choose Qt?” are quite easy to find: because of the compatibility, the abundance of functions, the performance of C++, the availability of the source code, the documentation, the high-quality technical support and many other points. But the most important point left: Qt is successful because Programmers like it. And this project is nothing more than further evidence. Next time I program an application with an interactive GUI (Graphical User Interface), I will use Qt.

Nowadays, Qt is much more than a tool for user interfaces. On the education slides on the Qt website, it is written that, it can be used “now for everything: Databases, XML, WebKit, multimedia, networking, OpenGL, scripting, non-GUI, …” (QT 2011 L1, p.3). It is also important to note that the C++ framework allows bindings with other programming languages like Java or Python. Qt is organized in modules and all “have a common scheme and are built from the same API design ideas” (QT 2011 L1, p.4). We can see an overview in the schema below:

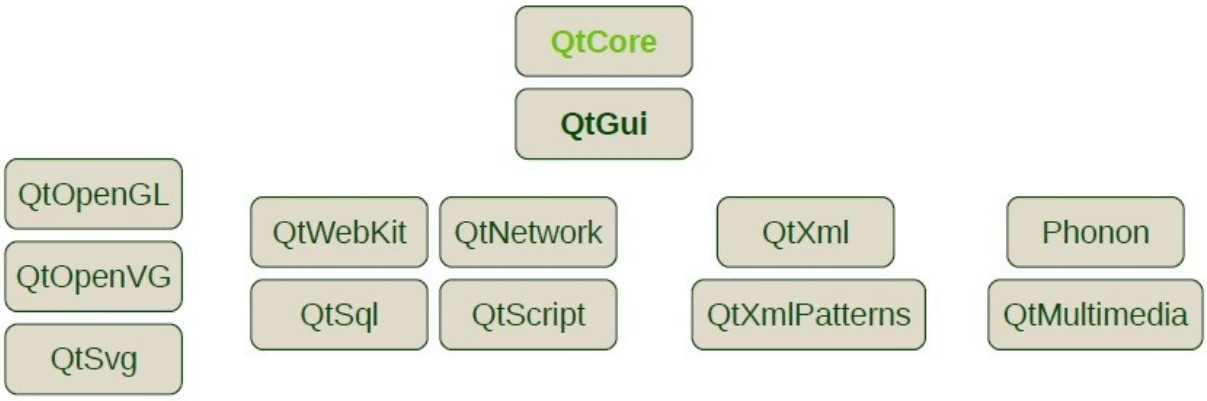


Illustration 18: Qt modules (QT 2011 L1, p. 4)

5.2.2 Qt Creator

Qt Creator is an application tailored to the needs of Qt developers. It's a cross-platform integrated development environment (IDE). It provides, among other things, a C++ code editor, project and build management tools, debuggers and an integrated UI designer.

The UI designer has become an essential feature. Historically, it was a separate tool, but is now a part of Qt Creator. The designer is simply a visual editor for forms; in other words, the designer allows to edit the display. It gives us the possibility to “play” with the in-/output interface: drag-and-drop widgets (buttons, tables, …), arrange layouts and make connections. Connections allow to “dynamically and loosely tie together events and state changes with reactions” (QT 2011 L2, p. 30).

More simply, they allow the user to interact with the display widgets and the system. For example, if the user click on a button, the system should react accordingly.

5.2.3 Compiler process

Another important matter about Qt is the compilation process. We can saw it summarized in the following schema:

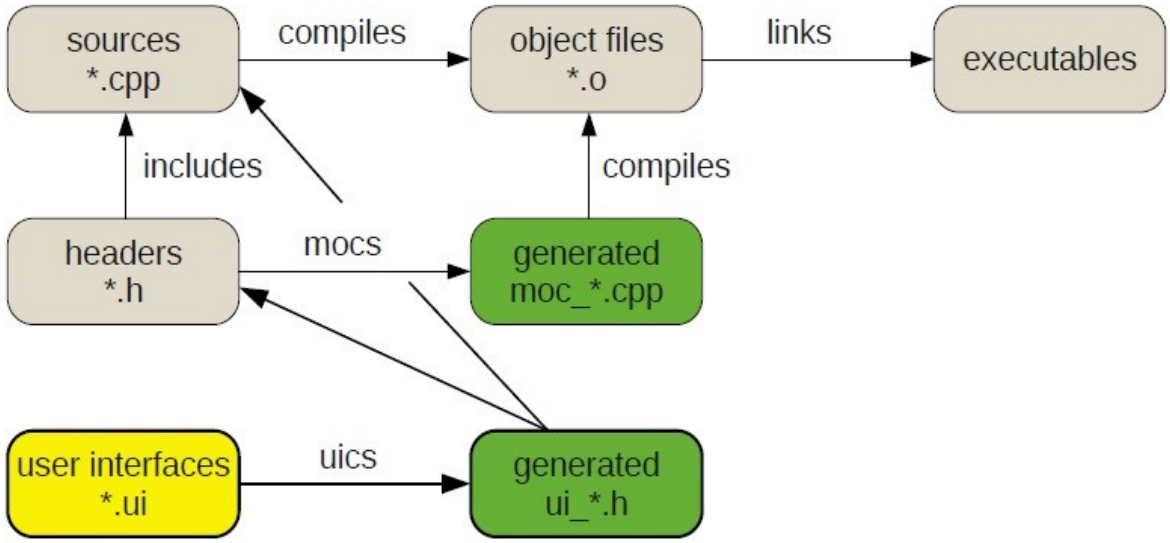


Illustration 19: Qt C++ build and compile process (QT 2011 L3, p. 40)

Gray labeled elements are present in the ordinary C++ build process., so that I am going to talk about the others. To begin, the “moc” (Meta Object Compiler) will be explained: it is the one that handles Qt's C++ extensions. After reading a C++ header file, if it finds one or more class declarations that contain the Q_OBJECT macro, it produces a C++ source file containing the meta-object code for those classes. Among other things, meta-object code is required for the signals and slots mechanism, the run-time type information, and the dynamic property system. The C++ source file generated by “moc” must be compiled and linked with the implementation of the class.

The second matter are the “user interfaces”. The UI designer automatically creates a file with the extension “.ui” which contains the forms description translated to XML (see the database explanation for more information about XML). From these files, the compiler creates header files and links them to other parts of the project. All together, source files, headers, user interface files and meta data combined, permit the creation of a program's executable file.

5.2.4 Signals and slots

The signals and slots mechanism permits to lie together events triggered by the user and system reactions. Let's take a look to the example illustrated in the following picture: each widget (buttons in this case) have different signals associated. In this concrete case, only the clicked() signal is going to be taken into account. By default, all the signals are active but not connected (when the user triggers a signal by clicking a button, nothing will happen because nothing is pending of the signal).

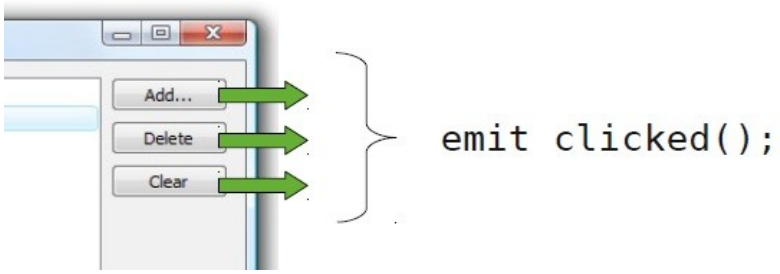


Illustration 20: Signals in a standard UI

The interesting thing is that with a simple code line, the signal can be connected with some kind of system reaction. The method is the following:

```
connect(Object1,SIGNAL(...),Object2,SLOT(...));
```

The programmer should indicate between which objects the connection should be done. Usually, "Object1" refers to the widget who emits the signal and "Object2" refers to the class defining the slot. A slot is implemented as any ordinary method and can be called as any ordinary method. Returning to the previous example, the clicked() signal from the buttons will be connected to the indicated private slots (check the picture below):

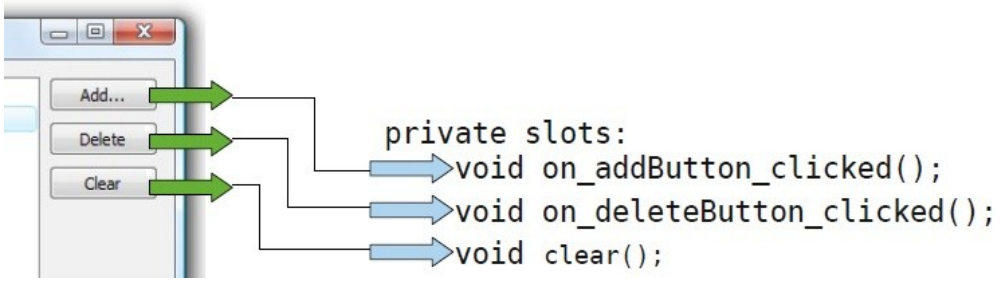


Illustration 21: Signals and slots connection in a standard UI

The code for this case will be ("this" stands for the class where the slots are defined):

```
connect(AddButton,SIGNAL(clicked()),this,SLOT(on_addButton_clicked()));
connect(DeleteButton,SIGNAL(clicked()),this,SLOT(on_deleteButton_clicked()));
connect(ClearButton,SIGNAL(clicked()),this,SLOT(on_clearButton_clicked()));
```

5.2.5 Our display

Finally, the main window of this project is presented (see picture below). The programm is organised in tabs and tool boxes (like tabs but vertically). We can count four tabs and each tab contains the features corresponding to the title. For example, “Time model” contains the features to add a new time model, to add a core period week or a week shift to a concrete time model and to change the attributes or to delete a time model.

The last three tabs (“Tree views”, “Calendar view” and “Roster view") are the output display of the programm. The “Tree views” tab allow the user to see the time models in the system and which users are linked to them. The “Calendar view” tab shows for a concrete day, what are the workers doing (working in a concrete week shift, on holidays, sick, …). Finally, the “Roster view” present workers ocupation on a concrete week. This display is very usefull to see the week shifts of the employees. For a simple walkthrough of the programm, check the appendix.

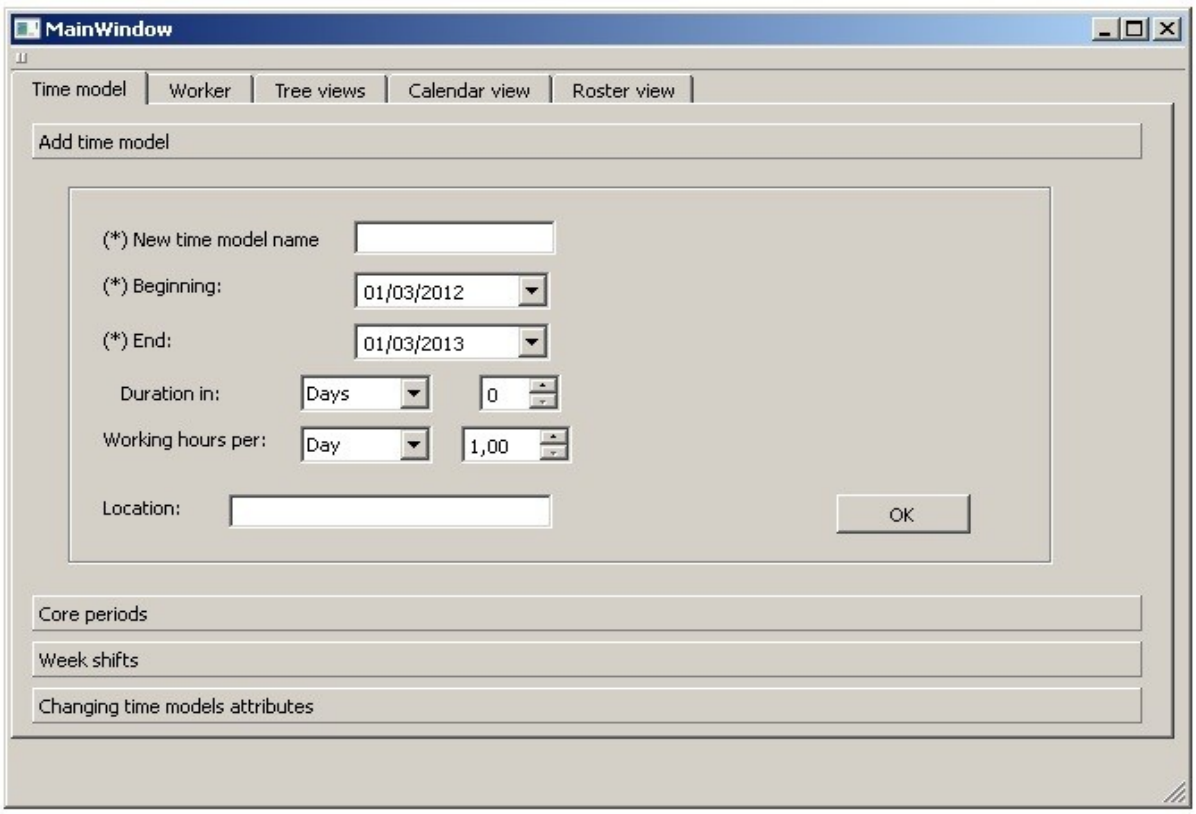


Illustration 22: MainWindow screenshot

5.3 Database

The database of this project is very simple: it consists in two XML documents. XML is the abbreviation for Extensible Markup Language. Furthermore, according to KEMPLER and EICKLER (2006, p.541), XML is the "new" language for the Web; it was conceived and standardized by the Word Wide Web Consortium (W3C). Actually, it is quite used as standard format for data exchange between distributed applications.

"XML documents are made up of storage units called entities; they all have content and are all (except for the document entity or root element) identified by the entity name." (BRAY, PAOLI, 2008). In fact, XML is very similar to HTML and the whole document is organized as a tree. According to VOSSEN (2008, p.338), from a formal point of view, an XML document is an ordered, hierarchical tree of elements".

For example, our workers' XML document (simplified):

```
<!DOCTYPE Wdata>
<workerdata>
 <worker holidaysKind="DaysPerYear" name="Victor" holidaysNum="20">
  <timemodel name="FullTimeCP"/>
  <holiday date="25122012"/>
  <holiday date="01012013"/>
  <overtime end="18:15" start="18:15" date="19042012"/>
 </worker>
 <worker holidaysKind="DaysPerYear" name="Alex" holidaysNum="20">
  <timemodel name="FullTime8h"/>
  <overtime end="00:00" start="00:00" date="20042012"/>
 </worker>
 <worker holidaysKind="DaysPerYear" name="Edward" holidaysNum="20">
  <holiday date="03082012"/>
  <holiday date="02082012"/>
  <holiday date="01082012"/>
  <overtime end="00:03" start="00:03" date="20042012"/>
 </worker>
 <worker holidaysKind="Education" name="Constance" holidaysNum="0">
  <timemodel name="e2"/>
  <holiday date="25122012"/>
  <holiday date="01012013"/>
  <illnessday date="01012012"/>
  <overtime end="00:34" start="00:34" date="20042012"/>
 </worker>
</workerdata>
```

The workers' XML can also be drawn as the following tree:

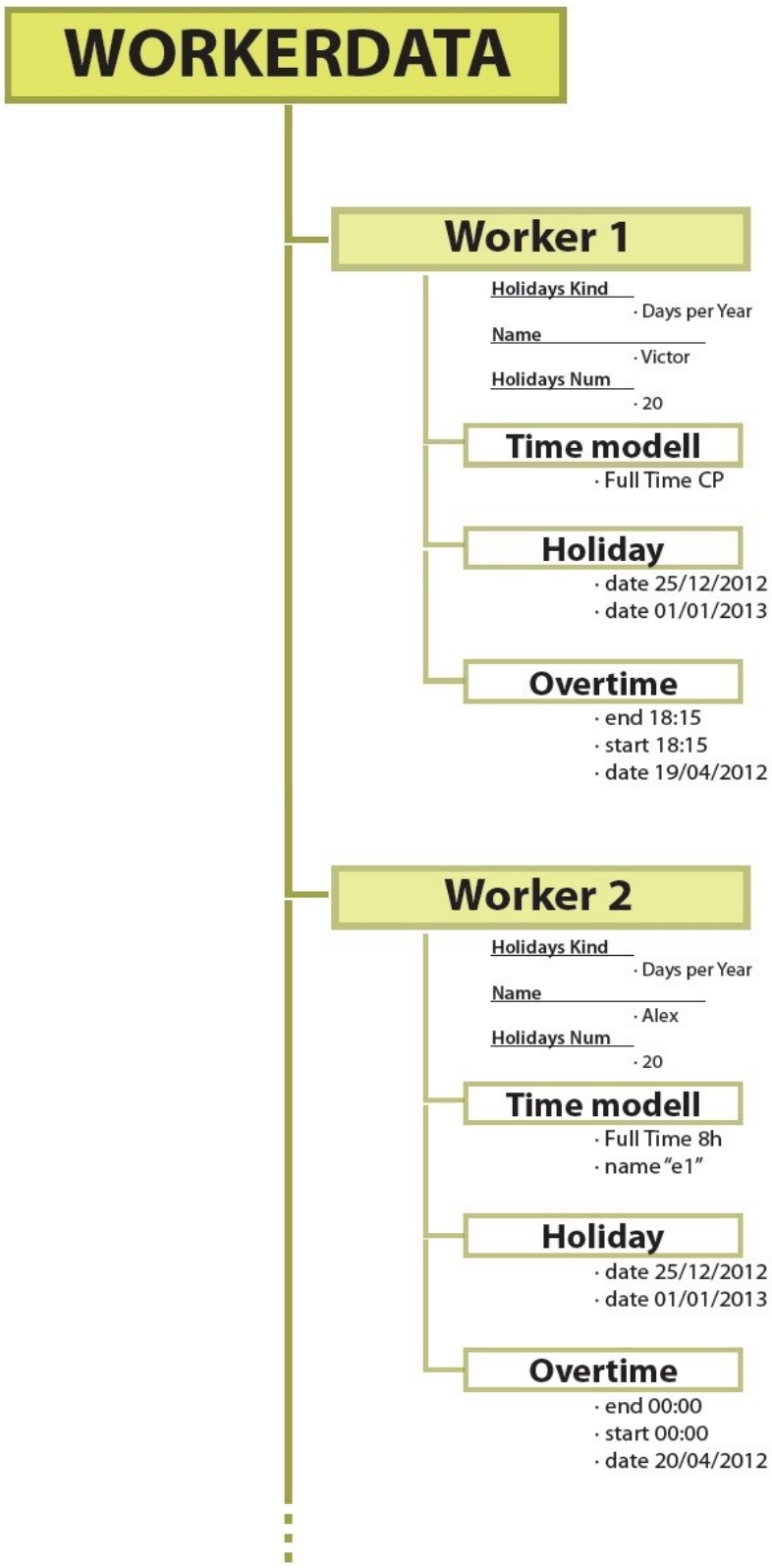


Illustration 23: Tree view of the workers' XML

“Several approaches to organizing the contents of XML documents to facilitate their […] retrieval have been proposed.” In this project I am going to use a “Database Management System (DBMS) to store the document as a text” (ELMASRI, NAVATHE, 2007, p.932). The DBMS consists in a special module (or object) called QdomDocument ant it is included in the standard Qt Library (starting from version 3.3). This object implements the Document Object Model (DOM), a language-neutral and platform-neutral way to access an XML document according to ANDERSON, BIRBECK, KAY and others (2000, p. 159).

In fact, a DOM document can also be seen as a tree. We can observe that every object from the XML generates a “QDomNode” organized as a tree. An example of a DOM-tree view can be found on the following picture:

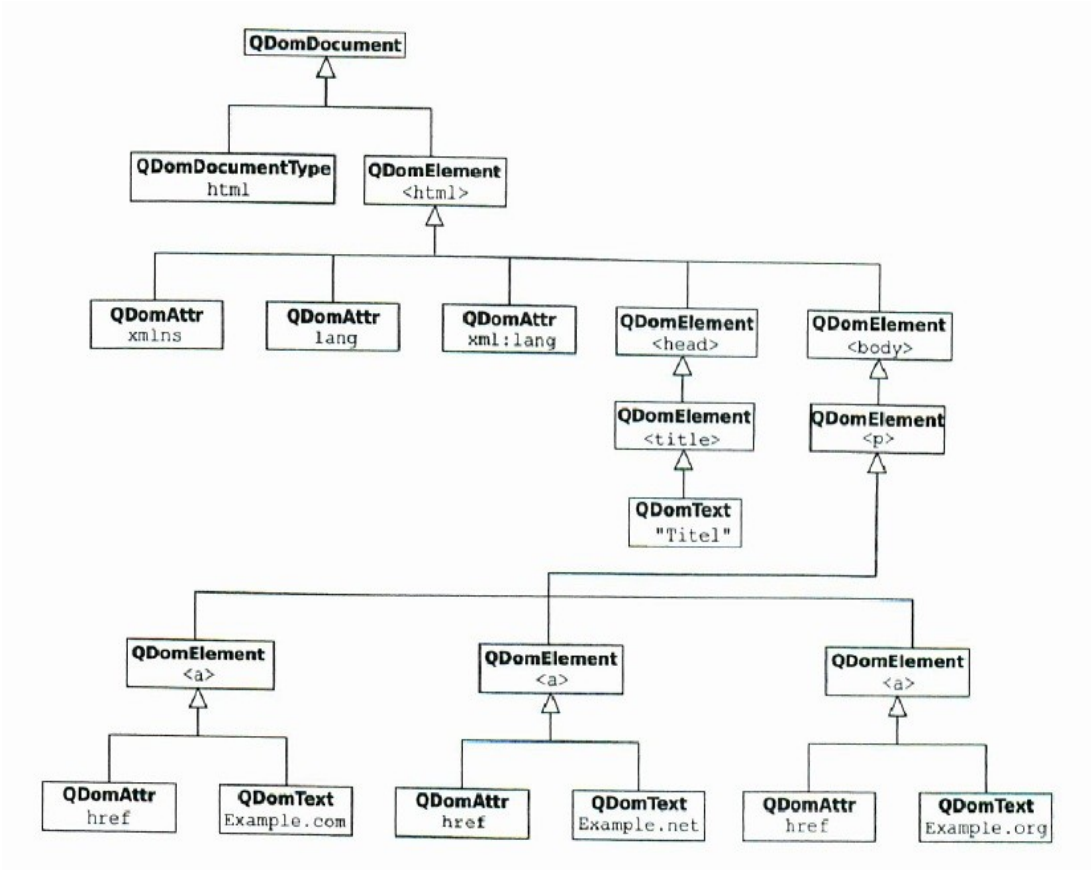


Illustration 24: DOM-tree view of an QDomDocument (MOLKENTIN 2006, p. 371)

Even if XML has some limitations, it fits perfectly the purpose of our work. For example, “XML is not optimized for access speed” (RAY, 2003, p. 331), but date base is not continuously accessed, only to save the document. Furthermore, “XML is not compact.” (RAY, 2003, p. 331). This means that the data will be stored in text files, but the data is not that big. XML documents can grow easily and become complicated to understand for normal persons but our data is very simple.

The way the program stores the XML document is nested. In other words, one class takes care of saving his data (attributes) and delegating the store work to each class within.

5.4 Deployment

The software furnished with this document is burned on a DVD. A ".exe"-file with the program ready to be executed on a Microsoft Windows environment will be found on the executable folder. As the program data is stored in XML files, the software should be executed on a folder with write permission, so the data could be correctly saved. The DVD has writing permissions, but the user's computer has to be able to write DVDs. A recomendation: the program should be copied to the local computer to use it.

If the user has not the Qt basic libraries installed, they can be found with the executable file. The user only has to execute the program on the same directory where those libraries are. The first time that the software is used, some error messages will show up indicating that the databases do not exist. This reaction is normal because the XML files are only created after introducing some workers or working time models in the system.

On the source code folder, the complete source code can be found. A good behavior would be to copy the files to a concrete workspace on the local computer and then open it with Qt creator in order to further develop the program. Any environment where Qt creator can be executed can be used to keep writing code.

6. Evaluation

In this chapter, an evaluation of the software will be made. A good software evaluation can be invaluable for many reasons, for both end-users and programmers. For example, when deciding and justifying enterprise software purchases or when making a more user-friendly program. Of course, there is cost in conducting these types of evaluations. But the results typically out weigh the costs by a large margin. Any large to medium sized organization can benefit from this approach.

Conducting a formal software evaluation in organizations can be very complicated. Issues like lack of understanding of the evaluated software, misunderstanding of requirements and others can be overcome by collaborating across IT and business departments. It is also very important to follow a concrete methodology: first, the requirements should be captured, ranked, and weighted. This should be done in an iteration process in order to increase the confidence that the correct decision will be made. Scoring may change from the previous iteration based on new or more information. If done correctly, this approach also brings the subject matter experts and technical gurus together to resolve their differences.

On the other hand, for programmers, a good software evaluation can also be a critical topic. For this reason, I am going to study the quality software concept from the point of view of the project and to have a look to some evaluation methods on the first part of the chapter. Then, a survey with some volunteers will be discussed.

6.1 Testing and developing

In this project, testing is done during implementation. In other words, during the development of the software, the designer used to test at the same time. Chronologically, the first implemented element was the time model and the “Add time model” tab in the user interface. Then, in order to test it, the “Changing time models attributes” was implemented. This tab allows us to check and change the attributes of a time model. After that, the core period and the week shifts were done.

On the same way, the worker element was implemented. First the “Add worker” tab, then “Changing workers attributes”, “Overtime”, “Holidays”. The “Illness days” tab was implemented later on because the sick leave was added subsequently. Once the workers and time models were finished, the implementation move on and started the visualization tabs.

The first and easiest part to implement was the “Tree view” tab. Although the “changing” tabs allowed to check modifications, the tree view was a critical point because the way the lists of elements are accessed is not the same. The tree view was implemented using a double loop (check the code for more information). Then came the “Calendar view” and the “Roster view” tabs. On the first, the user can watch the workers' occupation during a concrete day. The second one shows the week shift of some selected workers during a particular week. Somehow, all the visualization tabs mean to be software “tests”. A screen-shot of the software during the developing stage follows.

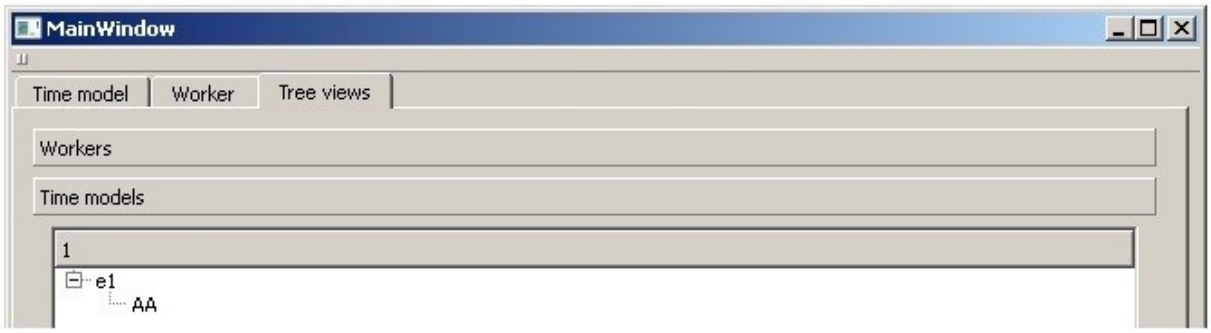


Illustration 25: Screen-shot of the program during the developing process

Before evaluating the software, it is also important to talk a bit of software quality and the degree to which it meets the specified requirements.

6.2 Software quality

“The ISO/IEC 9126 standard […] categorizes internal and external software quality characteristics” (SPINELLIS 2006, p. 4). This standard and the subsequent ISO 25000 (2005) quality model, also known as SquaRE, defines the the structure, classification and terminology of attributes and metrics applicable to software quality management. The Consortium for IT Software Quality (CISQ), an independent organization founded by the Software Engineering Institute (SEI) at Carnegie Mellon University, and the Object Management Group (OMG) defined the software quality characteristics according to those standards. For more information, check CISQ (2012).

According to OULD (1999, p. 171), software quality attributes are divided in two groups:
functional quality attributes and non-functional quality attributes. The first set refers to functional requirements and the second to non-functional requirements. As said in the upper section of his chapter, the functional requirements were tested during the implementation and at the end during the tests carried out by volunteers (check next section for more information). Only remains to discuss the non-functional requirements.

In chapter 4, four main non-functional requirements were decided:

- Reliability (the software executes without failure or for a specific period of time);
- Robustness (the degree to which a system continues to functions properly when confronted with invalid outputs, defects, or unexpected operating conditions);
- Usability (how easy it is for user to use or learn to use the software);
- Re-usability (how easy it is for programmers to extend the software).

Reliability and robustness are related and both were taken into account during the implementation. According to MAXION and OLSZEWSKI (1998, p. 346), "Programs fail mainly for two reasons: logic errors in the code, and exception failures". Because of that, the code has exceptions handling and the code logic was double checked before the final release. Of course, the software can still have errors but at least, they are minimized.

The re-usability requirement is included in the maintainability requirement. In the ISO/IEC 14764 (2006), four categories of maintenance are presented:

- Corrective: To correct discovered problems.
- Adaptive: To keep a software product usable in a changed or changing environment.
- Perfective: To improve performance, maintainability or to add new functionalities.
- Preventive: To detect and correct latent faults in the software product before they become effective faults.

Of course all these modifications are done after delivery. As said before, in this project, the re-usability is a critical requirement. Because of that, for example, the program uses standard Qt classes like "QString" and "QDate" instead of implementing own classes.

Usability has been defined in many different ways. The International Organization for Standardization's (ISO 9241, 1998) definition of usability is "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use." NIELSEN (1993, p. 26) divides usability into five measurable properties, which are commonly referred as usability attributes:

- Learnability: the software is quick and easy to learn,
- Efficiency: the software is efficient to use,
- Errors: the software is allows rapid recovery from errors,
- Satisfaction: the software is pleasant to use, and

- Memorability: the software is easy to remember.

Usability is often a crucial factor and because of that this project made a survey with some volunteers in order to evaluate the program.

6.3 Qualitative survey with volunteers

According to HEGNER (2003, p. 9), each type of evaluation seeks to pursue a given target. The objectives pursued through evaluations can be classified as follows:

- Comparative: at least two different systems are compared.
- Assessing: a certain required system property is checked.
- Analyzing: obtain information on vulnerabilities to propose solutions directly.

Comparing the data from the research done in chapter 2 about other existing commercial software and our program a couple of differences arise. The first distinction is the size of the applications: this project mean to be a simple and easy tool for working time modeling. Compared to other software, this program remains smaller, more efficient, faster and consumes less resources.

For assessing and analyzing the program, a survey was done with some volunteers. In order to allow the users to become familiar with the software, a quick tutorial with a time model examples was written (it can be found in the Annex). “Perhaps the simplest way to evaluate the usability of a design is to expose prospective users to the new product and solicit their opinions on various aspects of the design”(ARMSTRONG, BREWER and STEINBERG, 2002, p. 405). To do that, five volunteers who never used the program tried to follow the walkthrough (check the Annex) to draw up a 3-shift working time model with 3 workers. The users could also try any other functionality of the program.

The survey is included in the Annex with the results. The survey is extracted from STOWASSER (2006, p. 419) from the IFAB and focus on usability (this is the main reason to use it). A pair of conclusions can be drawn from those answers: the personalization level of the program could be improved; the error messages should be more explicit and helpful for the user; and the amount of information available to the user might be augmented. These three concepts can be included in the future prospects part.

7. Summary, conclusions and future prospects

In this last chapter, the summary and the conclusions of the thesis will be presented in the first part. Then, the future prospects for the project will be drawn. This last part is very important because the project mean to be a data model, a "container" for data and future extensions.

7.1 Summary and conclusions

Work has been characterized as the period of time spend at paid occupational labor and it holds an important place in the overall individual lifestyles. Therefore working related topics like working time models are interesting. This project has been an attempt to give an object-oriented data model for the existing working time models and rosters.

In doing so, this paper first investigated literature in the field of data models and software related to the main topic of this thesis. Two major conclusions can be drawn from this research. First, that the existing software leave a niche market for any new products. Second, by examining working time models as the contract between employers and employees, they had revealed much more than that. Actually, working time models determinate the relationship between workers and employers and had become an essential part in order to motivate and retain good workers. It's no longer a "paper to sign up", but an engagement of both parties and both should acknowledge it, specially the organizations.

The world today is constantly changing and evolving. Working time models already have an important role in creating or even reflecting this character by adding flexibility and creating new patterns around it. This project takes into account all these aspects and try to be flexible and loose in setting a working time model.

To include the stakeholders (workers, employers, companies, governments, …) in the design process can prove to be most illuminating both for the designer and the users themselves. In short, the goal is to create user-involved solutions. It seems, however that programming maintains a dominant position respect other user-friendly requirements and the user becomes a passive observer and consumer. The tools to avoid such an outcome exist and that's where the evaluation chapter plays an important role.

It is also important to note the technologies used in this diploma thesis. The two most important are Qt and C++. The first is a very useful framework for developing user interfaces and I think that it is a very interesting finding. The second is probably one of the most powerful and used programming languages of the world. Although I already programmed some code, I think that this project has allowed me to improve my understanding of the language.

7.2 Future prospects

The major contribution to knowledge that this thesis has made is the data model. This model should be understand as a technological base for future extensions because its only functionalities are the data storage and consultation. The amount of possible relevant extensions is only limited by our imagination.

For the future projects, it's important to say that, even if I tried to summarize the existing working time models in chapter 3, there is too many to fit in only one master thesis. A further research could be made on this field. It is also important to note that working time models also evolve. Maybe in the next years, a normal full-time working time model is no longer used.

Also important for future projects, is what kind of enterprise use the program. This thesis tried to make an model independent from the companies but the software developed is pretend to be used by those organizations. That means that it could be interesting to make some collaboration programs or trainings with the firms using the program in order to improve it and even develop it to fit some particular requirements. Workshops can provide insightful vision and brainstorm sessions are a valuable tool to glean intuitive information from the users' perspectives. To summarize, more user-involved initiatives should be done in the future.

More specifically, the software could be expanded with workstations. A workstation mean to be an extension of the location. Particularly, a worker could be assigned to a workstation situated in a concrete location. The employee works no longer in a particular country or city, but works on a determinate workstation. For example, in the case of industry, an operator in an assembly line would be responsible for using a welding machine.

It would also be interesting to add some control procedures to the software. For example, if a time model begins the 5 Mai 2012 ad ends the 5 Mai 2014, its duration should be updated automatically to 2 years or 24 months and vice-versa. This kind of procedures were not implemented in this

project because they can only be interesting when implementing functionalities using the data on the duration.

Another thing that could be implemented are the workers and employers preferences. Through this, the software could be personalized for each person according to the individual requirements like holidays, duration of breaks or starting and ending times. Even the place of work could be personalized. A possible implementation would be to add a preferences class containing all the information associated to a concrete worker. Once this job is done, the designers could use all this material to add more software funtionalities. For example, optimizing rosters and schedules in function of workers' preferences.

On the field of existing code field, it would be interesting to optimize some parts of the code. For example, workers' “timeModelsList” should be a list of pointers to the time model instead of a list of strings. Actually, the worker has a list of strings with the time models name and each time the system needs a concrete time model information from the time models list, a search has to be done. It would be much more easy to have pointers to the time models.

Another possible improvement could be to enlarge the letters and signs accepted by the software. That would improve the individualization of the project for each country or region. Actually, the program is though to work with regular English. That means for example that letters like Ü or Ñ are not accepted. This kind of cultural adaptation is highly valued by users.

8. References

8.1 Bibliography

ACAS

Booklet: Flexible working and work-life balance

London, Advisory Conciliation and Arbitration Service, September 2010

http://www.acas.org.uk/CHttpHandler.ashx?id=661&p=0 (visited 10/05/2012)

AINSWORTH, Susan ; PRUSS, Alice :

Same time, next year?: Human resource management and seasonal workers

In: Personnel Review, Vol. 38 Iss: 3, pp. 217 – 235

United Kingdom: Emerald Group Publishing Limited (2009)

http://www.emeraldinsight.com/journals.htm?articleid=1784687 (visited 11/05/2012)

ALEXANDER, Rosie :

Flexible working hours

AGCAS, Association of Graduate Careers Advisory Services

http://www.prospects.ac.uk/flexible_working_hours.htm. (visited: 10/05/2012)

ALHIR, Sinan Si :

UML in a nutshell

A desktop quick reference

Sebastopol (U.S.A.): O'Reilly & Associates, Inc. 1st Edition 1998, p. ix

ARMSTRONG, Stephen D. ; BREWER, William C. ; STEINBERG, Richard K. :

Usability Testing

In: Handbook of Human Factors Testing and Evaluation, chapter 18, p. 403-432

Editors: CHARLTON, Samuel G. and O'BRIEN, Thomas G.

New Jersey (U.S.A.): Lawrence Erlbaum Assoc. Inc., 2nd edition, 2002

ANDERSON, R. ; BIRBECK, M. ; KAY, M. and others :
XML Professionell
Translated by PAUL, E. ; JAEKEL, B. ; JAEKEL, U. ; ENGEL, R.
Bonn: MITP Verlag, 1. Auflage 2000 p.159

BECKER, M. ; EISSSING, G. ; EULER, H. P. and others :
Kompedium der Arbeitswissenschaft
Editors: HETTINGER, Th. ; WOBBE, G.
Ludwigshafen (Rhein): Friedrich Kiehl Verlag GmbH 1993

BEERS, Thomas M. :
Flexible schedules and shift-work: replacing the “9-to5” workday?
In: Monthly Labor Review Online, June 2000, Vol. 123, No. 6, p. 33
U.S. Department of Labor (Bureau of Labor Statistics)
http://www.bls.gov/opub/mlr/2000/06/art3full.pdf (visited 13/05/2012)

BLANCHETTE, Jasmin ; SUMMERFIELD, Mark :
C++ GUI Programmierung mit Qt 4
München: Addison-Wesley Verlag, 2., aktualisierte Auflage 2009

BLYTON, Paul :
Changes in Working Time: An International Review
London, Sydney: Croom Helm Ltd. , 1985

BMW :
Flexible working hours at the BMW Group
München, Bayerische Motoren Werke AG, February 2002

http://www.bmwgroup.com/e/0_0_www_bmwgroup_com/unternehmen/publikationen/aktuelles_lexikon/_pdf/FlexWork_E.pdf (visited 12/05/2012)

BÖKER, Karl-Hermann :

Spezielle Software für „Gute Schichtarbeit“: Schichtplan-Modellierungs- und Bewertungs-Software im Vergleichstest ; BR + PR Digital

In: Computer und Arbeit, Vernetztes Wissen für Betriebs- und Personalräte 1/2011, 28-32

Frankfurt am Main: Bund-Verlag GmbH, 2011

BRAY, T. ; PAOLI, J. ; SPERBERG-MCQUEEN, C.M. and others :

Extensible Markup Language (XML) 1.0

W3C Recomendation 26 November 2008

Firth Edition: http://www.w3.org/TR/xml/ visited: 25/04/2012

BRAINARD, Keith :

Phased Retirement Overview: Summary of Research and Practices

For the NASRA (National Association of State Retirement Administrators), october 2002

http://www.nasra.org/resources/Phased%20Retirement%20Overview.pdf (visited 11/05/2012)

BRENNAN, Clare :

Term-time working

iVillage.co.uk 2012, p. 1

http://www.ivillage.co.uk/term-time-working/83046 (visited: 10/12/2012)

BROWN, Marie :

Part Time Vs. Full Time Hours

eHow, visited 09/05/2012

http://www.ehow.com/about_5388085_part-vs-full-time-hours.html

BURKE, Ronald J. :

Research Companion to Working Time And Work Addiction

Massachusetts(U.S.A.): Edward Elgar Publishing, 2007

CARL, Andrea-Hilla ; MAIER, Friederike :

Flexible working time arrangements in Germany

In: External report commissioned by and presented to the EU Directorate-General Employment and Social Affairs

Berlin, March 2009

http://www.fgb-egge.it/public/documets/Flexible%20working%20time%20arrangements%20in%20Germany.pdf , visited: 12/03/2012

CARTER, Jean :

Voluntary reduced working time policy

Salford City Council, Human Resources, Organisational Development & Equality Group

http://www.salford.gov.uk/workingtime.htm (visited 12/05/2012)

CHRISTIANSEN, K. E. ; STAINES G. L. :

Flextime: A viable Solution to work/family conflict?

In: Journal of Family Issues, December 1990 vol. 11 no. 4 p. 455-476

London: SAGE Publications Ltd. 1990

CISQ :

Consortium for IT Software Quality website, 2012

http://www.it-cisq.org/ (visited 29/05/2012)

COSTAL COSTA, D. ; SANCHO SAMSÓ, M. R. ; TENIENTE LÓPEZ, E. :

Especificación de sistemas software en UML

Barcelona: Edicions UPC, 1era edición, noviembre 2003

COSTA, Giovanni :

Factors influencing health of workers and tolerance to shift-work.

In: Vol. 4, no. 3-4, Theoretical Issues in Ergonomics Science

London: Taylor & Francis 2003, p.264

COSTER, Helen :

How to take a sabbatical year from work

Forbes.com 24/08/2010

http://www.forbes.com/2010/08/24/sabbatical-leave-work-leadership-careers-advice.html (visited 10/05/2012)

CVTIPS :

What are the disadvantages of shift work ?

http://www.cvtips.com/career-success/what-are-the-disadvantages-of-shift-work-.html (visited 15/05/2012)

DEITEL, P. J. ; DEITEL, H. M: :

C++ How to Program

Upper Saddle River, New Jersey (U.S.A.): Pearson Education, Inc. , 6th Edition, 2008

ELLIS-CHRISTENSEN, Tricia :

Article: What is a Seasonal Job?

Edited by: WALLACE, O.

http://www.wisegeek.com/what-is-a-seasonal-job.htm (visited 11/05/2012)

ELMASRI, Ramez ; NAVATHE, Shamkant B. :

Fundamentals of Database Systems

Boston (U.S.A.): Pearson Addison Wesley, 5th edition 2007

ERIKSSON, Hans-Erik ; PENKER, Magnus :

UML Toolkit

New York (U.S.A.) : John Wiley & Sons, Inc. , 1st Edition 1998

EVANS, John M. ; LIPPOLDT, Douglas C. and MARIANNA, Pascal :

Trends in Working Hours in OECD Countries

In: OECD Labour Market and Social Policy Occasional Papers, No. 45

OECD Publishing, 2001

http://dx.doi.org/10.1787/674061356827 (visited 29/05/2012)

FLEX PLANNER

What is Flexitime?

Wakefield, West Yorkshire (U.K.), Net Aspect Web Design, 2012

http://www.flexitimeplanner.com/Flexi-time-policy.aspx (visited 14/05/2012)

FOWLER, Martin :

UML konzentriert

Eine kompakte Einführung in die Standard-Objektmodellierungssprache

München: Addison-Wesley Verlag, 3.Auflage 2004

FORBRIG, Peter :

Objektorientierte Softwareentwicklung mit UML

München: Carl Hanser Verlag, 3. Auflage 2007

GAWO :

BASS 4 Demo-Powerpoint-Präsentation

Oldenburg, Gesellschaft für Arbeits-, Wirtschafts- und Organisationspsychologische Forschung e.V. , Mai 2010

http://www.gawo-ev.de/cms/uploads/Bass%204%20Demo%20Praesentation.pps (visited 30/05/2012)

BASS 4 Web: Kurzbeschreibung

Oldenburg, Gesellschaft für Arbeits-, Wirtschafts- und Organisationspsychologische Forschung e.V. , Mai 2012

http://www.gawo-ev.de/cms/index.php?page=bass-4 (visited 30/05/2012

HEATHFIELD, Susan M. :

about.com (Human resources guide), 2011

Shift-work: http://humanresources.about.com/od/glossarys/g/shift_work.htm visited: 28/01/2012

Advantages and disadvantages of flexible work schedules?

About.com guide to Human Resources

http://humanresources.about.com/od/employeebenefits/f/flex_schedules.htm (visited 13/02/2012)

HEGNER, Marcus :

Methoden zur Evaluation von Software

Bonn: Informationszentrum Sozialwissenschaften der Arbeitsgemeinschaft Sozialwissenschaftlicher Institute e.V. , Mai 2003

http://www.gesis.org/fileadmin/upload/forschung/publikationen/gesis_reihen/iz_arbeitsberichte/ab_29.pdf (visited 02/06/2012)

HÖFER, K: ; HOLZHÄUSER, T. ; LENNINGS, F. and others :

e-Shift-Design – Die Praxishilfe zur Schichtplangestaltung

Zeitschrift angewandte Arbeitswissenschaft des ifaa, Ausgabe 199, 2009.

e-Shift-Design: http://www.arbeitswissenschaft.net/Artikel-e-Shift-Design.774.0.html

visited: 27/04/2012

HOFFMANN, Dirk W. :

Software-Qualität

Berlin, Heildelberg: Springer-Verlag, 2008

HORNBERGER, S. ; KNAUTH, P. :

Innovative Flexibilisierung der Arbeitszeit.

In: Innovatives Arbeitszeitmanagement 2000.

Aachen: Shaker Verlag, 2000, p. 24

(ifab-Forschungsberichte aus dem Institut für Arbeitswissenschaft und Betriebsorganisation der Universität Karlsruhe; Band 22)

HUMPHREY, Andy :

The Disadvantages of a Compressed Work Week

eHow.com ; visited 13/05/2012.

http://www.ehow.com/list_6039423_disadvantages-compressed-work-week.html

IEEE Std 830 :

IEEE Recommended Practice for Software Requirements Specifications

IEEE Computer Society, 20 October 1998

http://standards.ieee.org/findstds/standard/830-1998.html (visited 26/05/2012)

ILO:

Fact sheet: Part-time work

Geneva, International Labour Office, May 2004

http://www.ilo.org/wcmsp5/groups/public/---ed_protect/---protrav/---travail/documents/publication/wcms_170717.pdf (visited 09/05/2012)

Annualized hours (hours-averaging) schemes

Information Sheet No. WT-12, May 2004

Geneva, International Labour Office - Conditions of Work and Employment Programme

http://www.ilo.org/wcmsp5/groups/public/---ed_protect/---protrav/---travail/documents/publication/wcms_170706.pdf (visited 14/05/2012)

ILO Declaration on Social Justice for a Fair Globalization

adopted by the International Labour Conference at its Ninety-seventh Session, Geneva, 10 June 2008

Geneva: ILO Publications, 2008

http://www.ilo.org/wcmsp5/groups/public/---dgreports/---cabinet/documents/publication/wcms_099766.pdf (visited 29/05/2012)

ISERNHAGEN, Rolf ; HARTMUT, Helmke :

Softwaretechnik in C und C++

München, Wien: Carl Hanser Verlag, 4. Auflage 2004

ISO/IEC 9126 :

Software engineering - Product quality

International Organization for Standarization, 2011

http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22749 (visited 01/06/2012)

ISO 9241:

Ergonomic requirements for office work with visual display terminals (VDTs)

Part 11: Guidance on usability

International Organization for Standarization, 1998

http://www.iso.org/iso/catalogue_detail.htm?csnumber=16883 (visited 01/06/2012)

ISO/IEC 14764 :

Software Engineering - Software Life Cycle Processes – Maintenance

International Organization for Standarization, 2006

http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39064 (visited 01/06/2012)

ISO/IEC 25000 :

Software engineering — Software product Quality Requirements and Evaluation (SquaRE)

International Organization for Standarization, 2005

http://webstore.iec.ch/preview/info_isoiec25000%7Bed1.0%7Den.pdf (visited 01/06/2012)

IWH :

Fact sheet – shift-work

Toronto, Institute for Work & Health, August 2005

http://www.iwh.on.ca/media/images/Shiftwork.pdf (visited 28/01/2012)

KATEPOO, Pat :

Pros and Cons of a Compressed Workweek as a Flexible Work Arrangement

In: Work Options 2012

http://www.workoptions.com/compressed-workweek-pros-and-cons (visited 14/05/2012)

KEMPLER, Alfons ; EICKLER, André :

Datenbanksysteme, eine Einführung

München: Oldenbourg, 6. Auflage 2006

LEE, Richard C. ; TEPFENHART, William M. :

UML and C++

A practical guide to object-oriented development

New Jersey (U.S.A.): Prentice-Hall, Inc. 1st Edition 1997

LEE, Sangheon ; MCCANN, Deirdre ; MESSENGER, Jon C. :
Working Time Around the World
Taylor & Francis e-Library, 2007

http://www.ilo.org/wcmsp5/groups/public/@dgreports/@dcomm/@publ/documents/publication/wcms_104895.pdf (visited 04/06/2012)

LEWIS, Christine :
Report: Taking the lid off term-time working in education
London: UNISON (www.unison.org.uk), December 2002

LONSDALE, Maggie :
Time Off in Lieu When Working Additional Hours
WorkingRights website, 2012
http://www.workingrights.co.uk/time-off-lieu-working-additional-hours.html (visited 03/06/2012)

MA, Li ; XU, Congwei :
Work-life conflict and the predictor of time-related job characteristcs:self-efficacy as a moderating factor
For IEEE: Institute of Electrical and Electronics Engineers, 2009
http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=05302062 (visited 18/05/2012)

MARSHALL, Katherine
Job sharing
In: Perspectives on Labour and Income, 1997, vol. 9 no. 2
For: Statistics Canada, Ottava (Ontario)
http://www.statcan.gc.ca/studies-etudes/75-001/archive/e-pdf/3069-eng.pdf (visited 12/05/2012)

MASCARENHAS, Pauline
Advantages of a Part Time Job in the Workplace
suite101.com , visited 09/05/2012
http://pauline-mascarenhas.suite101.com/advantages-of-part-time-jobs-in-the-workplace-a243203

MAXION, Roy. A. ; OLSZEWSKI, Robert T. :
Improving Software Robustness with Dependability Cases
28th International Symposium on Fault-Tolerant Computing: Munich, June 1998.
IEEE 1998, p. 346-355
http://www.cs.cmu.edu/~maxion/pubs/maxionolszewski98.pdf (visited 01/05/2012)

MOLKENTIN, Daniel :
Qt 4 Einführung in die Applikationsentwicklung
Editor: JUNG, Patricia
Copyright: Open Source Press, 2006
Krugzell: Kösel Verlag, 2006

MOSS, Richard L. ; CURTIS, Thomas D. :
Flextime and sick leave use
In: Atlantic Economic Journal, Vol. 12, Num. 2 (1984)
Publisher: Springer Netherlands, 1984

NIELSEN, Jakob :
Usability Engineering
San Diego (U.S.A.): Academic Press, Inc, 1993

NUBEY, N. B. :
Office management: developing skills for smooth functioning
New Dehli: Global India Publications Pvt Ltd. , 2009

OLIVER, Bruce ; CAPSHAW, Dan :
Employee Shift Work Schedules: An Introduction
Republished with permission from the Society of Human Resource Management
http://shift-work.com/articlesnewsletters/shiftwork-articles/employee-shift-work-schedules-an-introduction/ visited: 09/05/2012

OPM :

Report to the Congress: Status of Telework in the Federal Governement
United States Office of Personnel Management, 2010
http://www.telework.gov/Reports_and_Studies/Annual_Reports/2010teleworkreport.pdf (visited 11/05/2012)

Handbook on Alternative Work Schedules
United States Office of Personnel Management, 2012
http://www.opm.gov/oca/aws/html/flex.asp (visited 13/05/2012)

OUALLINE, Steve :

Practical C++ Programming
Editor: DENN, Robert.
Sebastopol, CA (U.S.A.): O'Reilly Media, Inc. , 2nd Edition 2003

OULD A. Martyn:

Managing Software Quality and Business Risk
Chischester (England): John Wiley & Sons, Ltd. 1999

PLANTENGA, Janneke ; REMERY, Chantal :

Flexible working time arrangements and gender equality: A comparative review of 30 European countries
For the European Comission (Directorate-General for Employment, social affairs and equal opportunities), November 2009
Luxembourg: Publications Office of the European Union, 2010

QT :

The ideas behind Qt and live demo
Qt in education course materials, slides L1
Nokia Corporation, 2011
http://qt.nokia.com/learning/education/course-materials (visited 23/05/2012)

The Qt object model and the signal slot concept
Qt in education course materials, slides L2
Nokia Corporation, 2011
http://qt.nokia.com/learning/education/course-materials (visited 23/05/2012)

Widgets and Layouts
Qt in education course materials, slides L3
Nokia Corporation, 2011
http://qt.nokia.com/learning/education/course-materials (visited 23/05/2012)

RAY, Eric T. :
Learning XML
Sebastopol (U.S.A.): O'Reilly & Associates, Inc. 2003, p. 345

RIFKIN, Jeremy :
The third industrial revolution
New York (U.S.A.): Palgrave Macmillan. 1st Edition, 2011

ROGAK, Lisa Angowski
Time off from work
New York, Chichester, Brisbane and others: John Wiley & Sons, Inc. 1994.

ROSA, R. R. ; COLLIGAN, M. J. :
Plain Language about shift-work
U.S. Department of health and human services 2007, p.6
http://www.cdc.gov/niosh/pdfs/97-145.pdf. 08/05/2012

RUMBAUGH, James ; JACOBSON, Ivar ; BOOCH, Grady :
The Unified Modeling Language reference manual
Massachusetts: Addison Wesley Longman, Inc. 1st Edition 1998

SAINT-CYR, Yosie :
Flexible work policies are a good business
For: Hrinfodesk.com - Canadian Payroll and Employment Law News, March 2007
http://www.hrinfodesk.com/preview.asp?article=21768 (visited 13/05/2012)

SHIFTWORK Ltd. :

Flexible working definitions

London, Shiftwork consultants Ltd.

http://www.swiftwork.com/flexible_working_definitions.asp (visited 10/05/2012)

SIMCHA, Ronen :

Alternative Work Schedules

Homewood, Illinois (U.S.A.): Dow Jones-Irwin, 1984, p. 116

SFEIR-YOUNIS, Alfredo :

Spirituality and public policy for Decent Work: Self-realization in the new millennium

In: Philosophical and Spiritual Perspectives on Decent Work

Editor: PECCOUD, Dominique

Geneva: ILO Publications, 2004

http://www.ilo.org/wcmsp5/groups/public/---dgreports/---exrel/documents/publication/wcms_175188.pdf (visited 29/05/2012)

SPINELLIS, Diomidis:

Code Quality: The Open Source Perspective

Massachusetts (U.S.A.): Addison Wesley, 2006

STOCK, Patricia ; ZÜLCH, Gert :

Reorganising the Working Time System of a Call-Centre with Personnel-oriented Simulation

In: Integrating Human Aspects in Production Management

For: IFIP (International Federation for Information Processing), Volume 160/2005

Editors: ZÜLCH, Gert ; JAGDEV, Harinder S. ; STOCK, Patricia

New York (U.S.A.): Springer Science+Business Media, Inc. , 2005

STOWASSER Sacha :

Methodische Grundlagen der softwareergonomischen Evaluationsforschung

In: Forschungsberichte aus dem Institut für Arbeitswissenschaft und Betriebsorganisation der Universität Karlsruhe, Band 37.

Editor: ZÜLCH, Gert

Aachen: Shaker Verlag GmbH, 2006

TELEWORK.GOV

Telework Managers

U.S.A. : the official website of the Federal Government's telework program

http://www.peogcs.army.mil/images/downloads/telework-managers.pdf (visited 11/05/2012)

VOSSEN, Gottfried :

Datenmodelle, Datenbanksprachen und Datenbankmanagementsysteme

München: Oldenbourg, 5. Auflage 2008

WIEGERS, Karl E. :

Software requirements

Redmond, Washington (U.S.A.): Microsoft Press, 2nd Edition 2003

WORKPLACE FLEXIBILY 2010 :

Flexible Work Arrangements: A Definition And Examples

Washingtin D.C. (U.S.A.): Georgetown University Law Center, 2010

Part of the Alfred P. Sloan Foundation’s National Initiative on Workplace Flexibility

http://workplaceflexibility2010.org/images/uploads/general_information/fwa_definitionsexamples.pdf (visited 16/05/2012)

XIMES:

Workbook: Time Intelligence Solutions

Wien: Ximes GmbH Austria, 2008

http://www.ximes.com/downloads/workbook_tis_de.pdf (visited 31/05/2012)

Diashow: Eine kleine Reise durch den Shift Plan Assistant in 15 Bildern

Wien: Ximes GmbH Austria, 2012

http://www.ximes.com/de/software/software/spa/spa_diashow.php (visited 31/05/2012)

ZOLA, Émile :

Germinal

La série des Rougon-Macquart: L'Histoire naturelle et sociale d'une famille sous le Second Empire

Saint Amand (Cher), France: Éditions Gallimard 1978, col. Folio Classique n. 3304, réédition 2001

ZÜLCH, Gert ; STOCK, Patricia ; HRDINA, Jan :

Working Time Configuration in Hospitals Using Personnel-oriented Simulation

In: Lean Business Systems and Beyond: First IFIP TC 5 Advanced Production Management Systems Conference (APMS'2006)

Editor: KOCH, Tomasz

New York (U.S.A.): Springer, 2008

Process Optimization and Efficient Personnel Employment in Hospitals

In: Operations Research Proceedings, 2007, Volume 2006, Part X

Editors: WALDMANN, Karl-Heinz ; STOCKER, Ulrike M.

Berlin, Heildelberg: Springer-Verlag, 2007

ZÜLCH, Gert ; STOCK, Patricia ; LEUPOLD, Michael :

Simulation-aided Design and Evaluation of Flexible Woring Times

In: Proceedings of the 2011 Winter Simulation Conference

Editors: JAIN, S. ; CREASEY, R.R. ; HIMMELSPACH, J. and others

www.informs-sim.org/wsc11papers/194.pdf (visited 30/05/2012)

8.2 Software

8.2.1 External Software

BASS 4 :

Version vom Dezember 2007

Dittmar & Schomann GbR, Oldenburg (Germany), 2008

e-Shift-Design :

Version 2.0

Rasselstein GmbH, Verband der Metall- und Elektroindustrie Rheinland-Rheinhessen e. V. (VEM) and the Institut für angewandte Arbeitswissenschaft e. V. (IfaA)

Düsseldorf (Germany), 2009

LibreOffice 3 :

Version 3.3.2

Based on OpenOffice.org

The Document Foundation, Berlin (Germany), 2010

Qt creator :

Version 2.4.1

Based on Qt 4.7.4 (32 bit)

Nokia Corporation, Espoo (Finnland), 2008-2011

Visual Paradigm for UML :

Community Edition

Version 8.3 (Build sp2_20120202)

Visual Paradigm International, Hong Kong, 1999-2012

Ximes OPA & SPA :

OPA (Operating Hours Assistant) : Version 4.0

SPA (Shift Plan Assistant) : Version 7.0

XIMES GmbH, Wien (Österreich), 2012

8.2.2 Internal Software

OSim-GAM :

Version 1.0

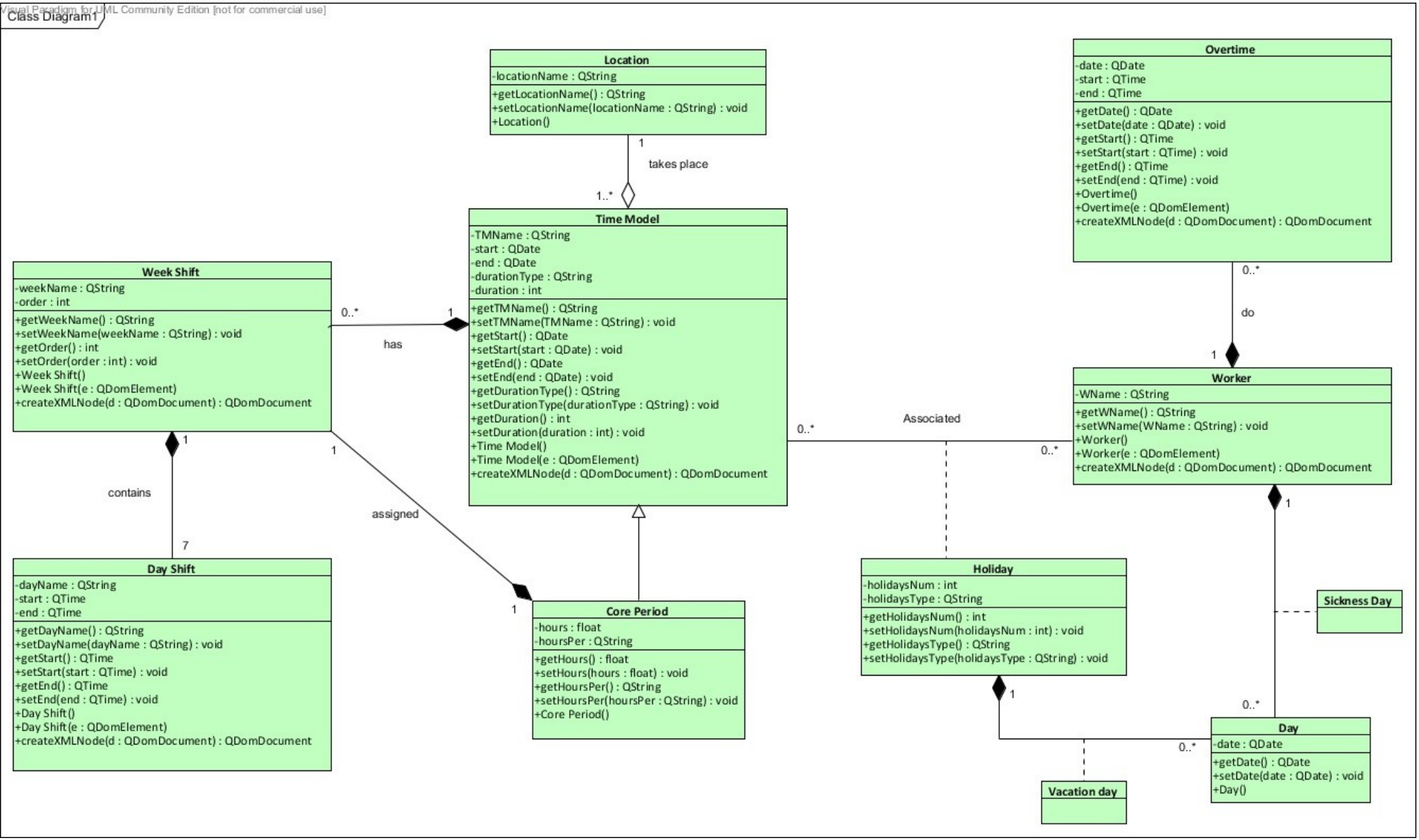
Institut für Arbeitswissenschaft und Betriebsorganisation (ifab)

Universität Karlsruhe 2004

9. Appendix A: UML Final Model

The final model developed during this project is presented in the next page of this appendix. This model contains all the attributes and functions required.

Class Diagram1



10. Appendix B: Walkthrough

In this appendix a little tutorial on the program is presented. In chapter 5 and chapter 6, some references are made to this appendix in order to better understand how the software is. We will draw up a 3-shift working time model with 3 workers (AA, BB and CC). The 3 shifts will be Fruhschicht (Morning shift), Abendschicht (Afternoon shift) and Nachtschicht (Night shift). The first one lasts from 08:00 until 16:00; the second one lasts from 16:00 until 00:00 and the night shift lasts from 00:00 until 08:00.

The first thing to to is creating 3 time models (one per each worker). Each working time model needs to have its own unique name, a beginning and an ending date. Optional parameters like the duration (in days, months or year), the working hours (per day, week, month or year) and the location can also be given. On the picture below, the Time model's tab of the program is presented and the dialog asking to confirm the input data for saving the time model.

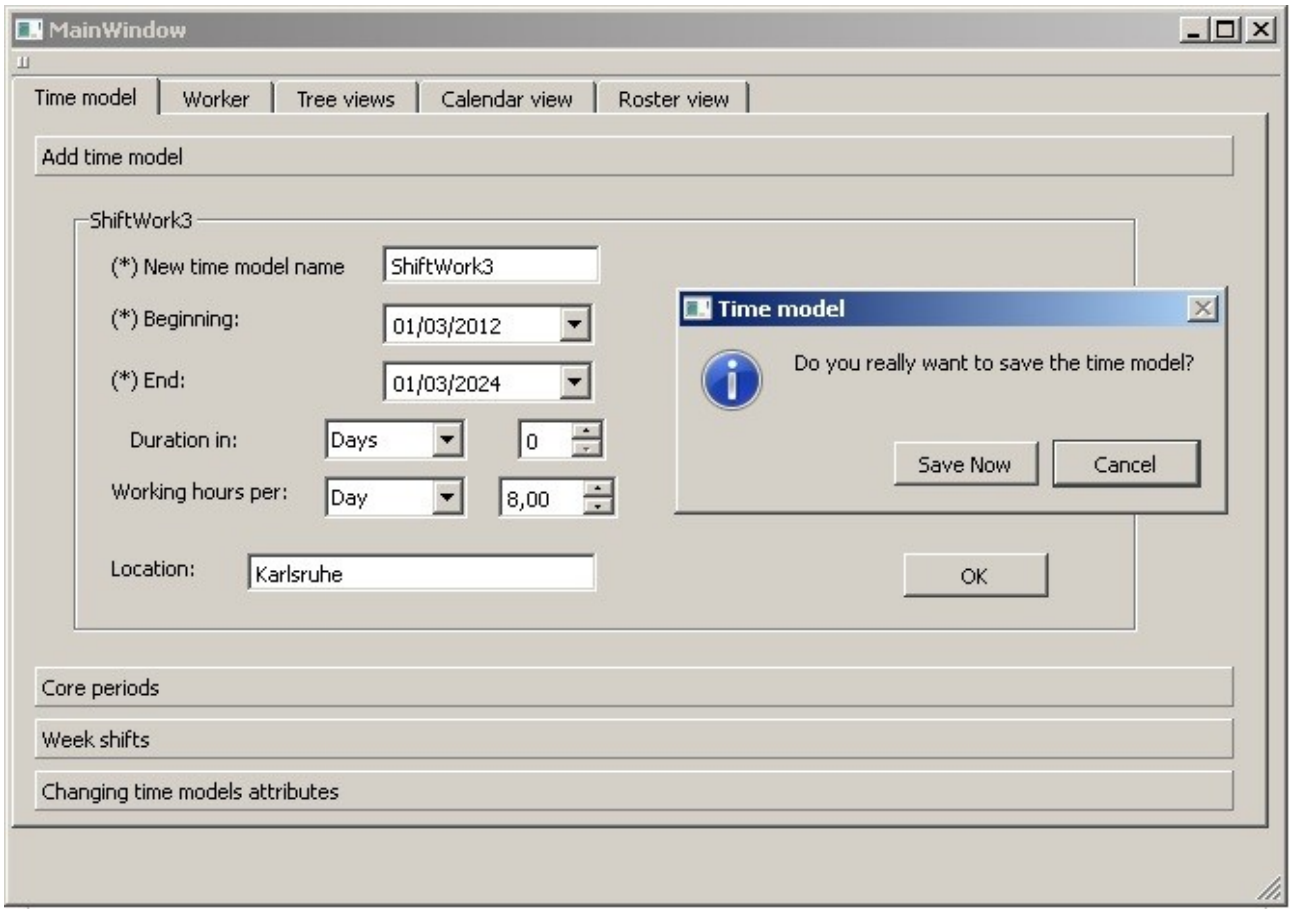


Illustration 26: Creating a time model on our program

The system also gives us the option to change the time model attributes in the lower tab. The system also give us the possibility to delete the time model. All the attributes can be changed except the time model name. The user is in charge that no unvalid data is introduced (for instance, the program does not check if the beginning date is posterior to the ending date). On the picture below, the location changed (from "Karlsruhe" to "Karlsruhe Oststadt").

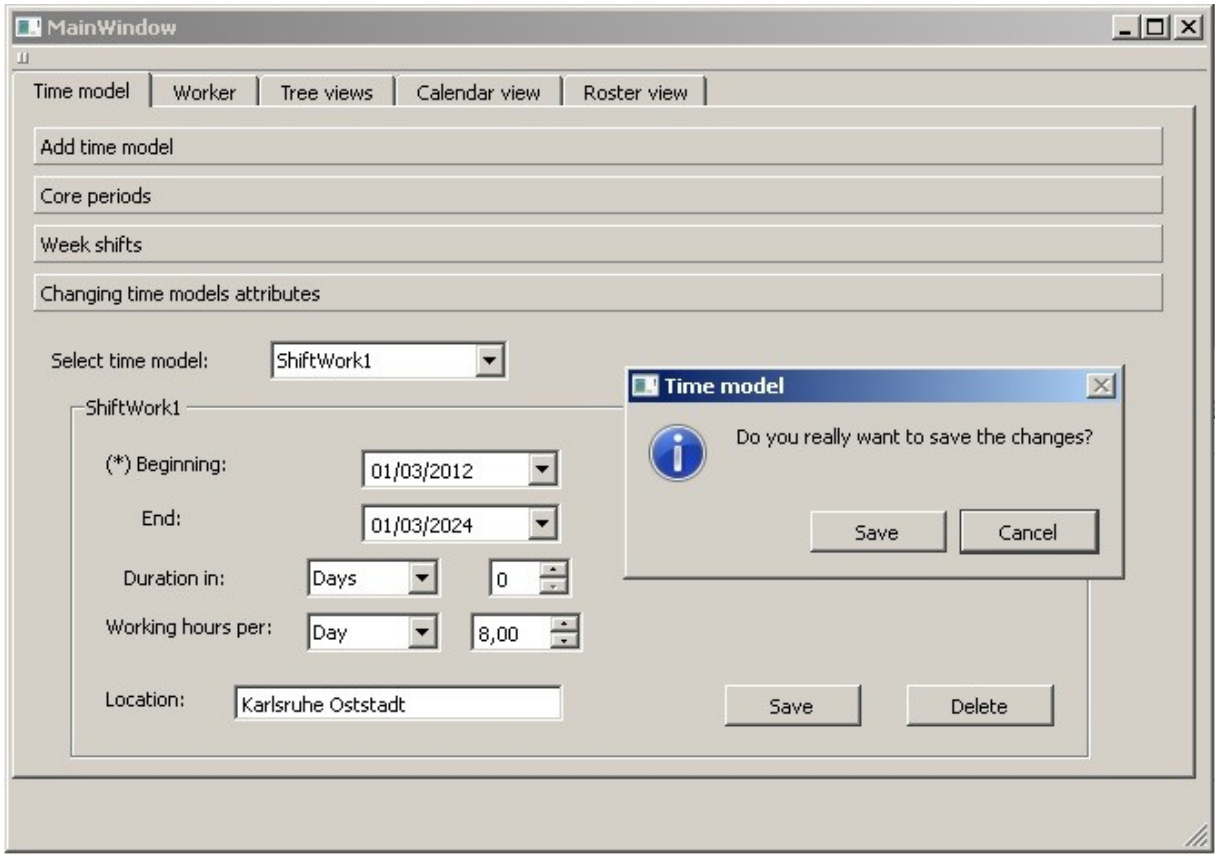


Illustration 27: Changing the time model location

Once the 3 different time models are created, they should be assigned to the correct week shifts. In order to do so, the corresponding week shifts will be created, like in the next picture:

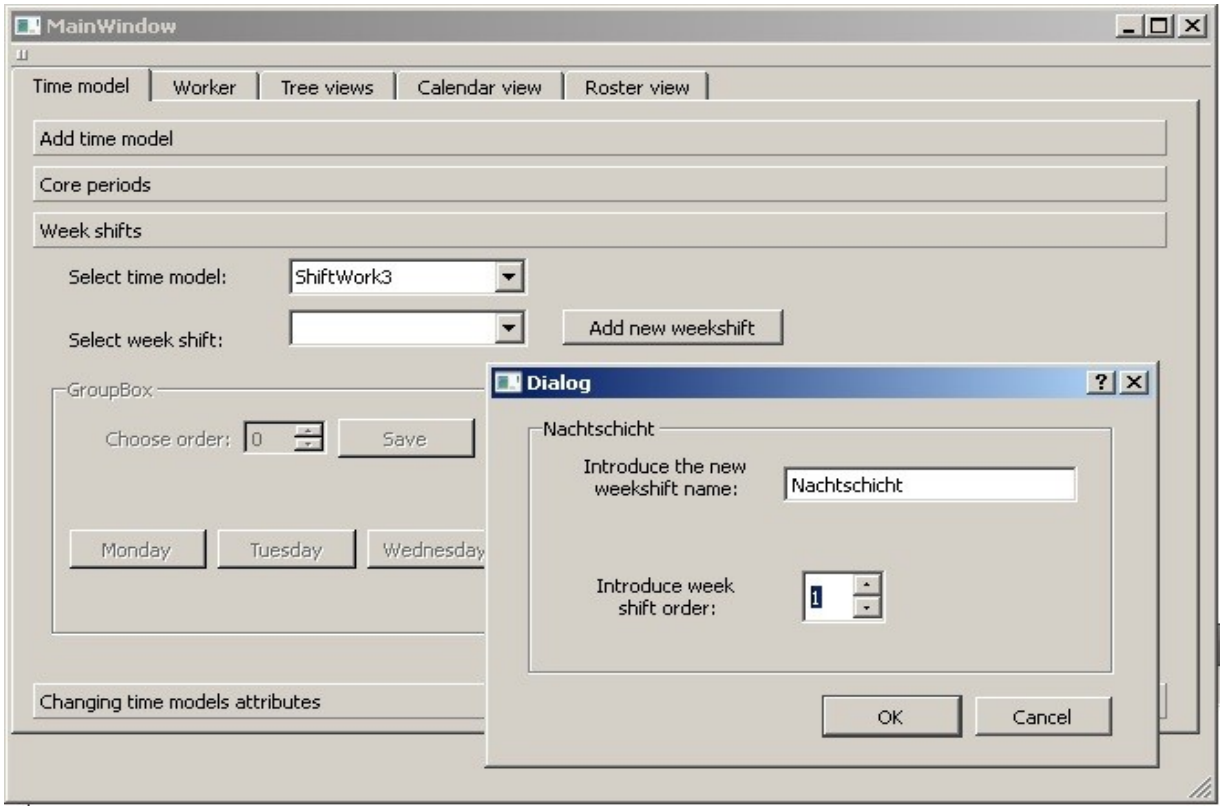


Illustration 28: Adding a new week shift to the time model

After selecting a time model, the system allows us to add a new week shift. To do so, a dialog pops up and asks us for a new name and an order number for the new weekshift. The order numbers are important because they indicate the system how should the week shifs be computed. In other words, if the worker has a week shift with order 1, he will start working with this week shift concrete schedule.

After that, the starting and ending times of each day of a selected shift should be introduced. By default all the day shifts are desactivated. On the picture below, the dialog for activating and setting up a concrete day shift is presented:

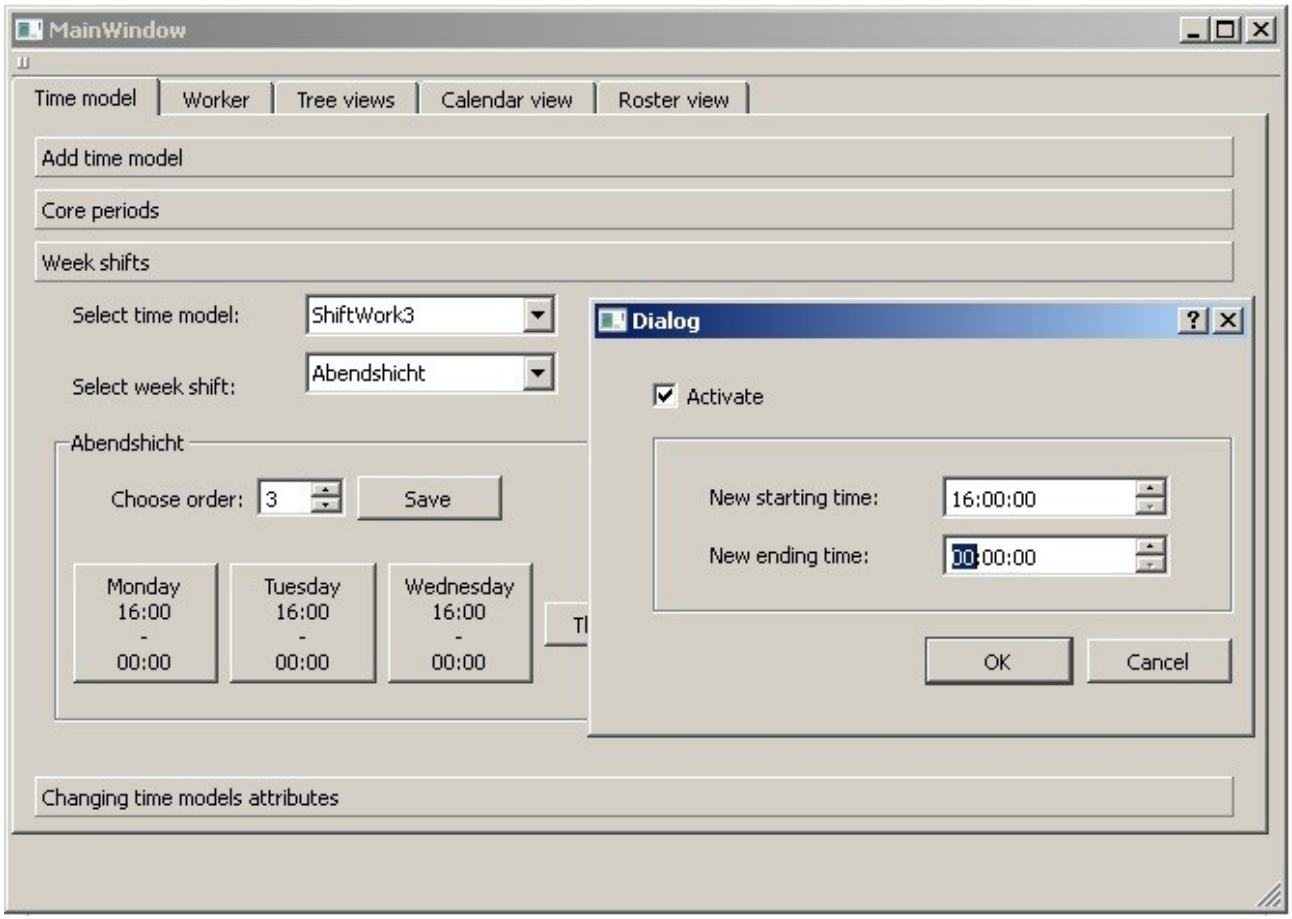


Illustration 29: Activating thrusday's shift

First of all, the concrete day shift should be activated (with the checkbox). Then, the starting and ending times introduced and finally accepted. Once a day is activated, the starting and ending times are shown in the week shift description. By contrast, only the day's name is displayed on the non-activated days.

The next step of this tutorial is the creation of workers. The picture below is a screenshot of this task (in the "Worker" tab of the main window). Creating a new worker is very easy: The system only needs a new valid name (with at least two letters and unique in the system), the holidays type (and maybe the number of vacation days per year) and a time model associated to the worker. All the attributes can be changed later except the worker's name. A pop-up dialog is shown before saving the worker to the system, so the user can check again all the data. A worker can have different time models associated and the user is responsible of maintaining the consistency. In other words, the user has to be sure that two different time models do not overlap.

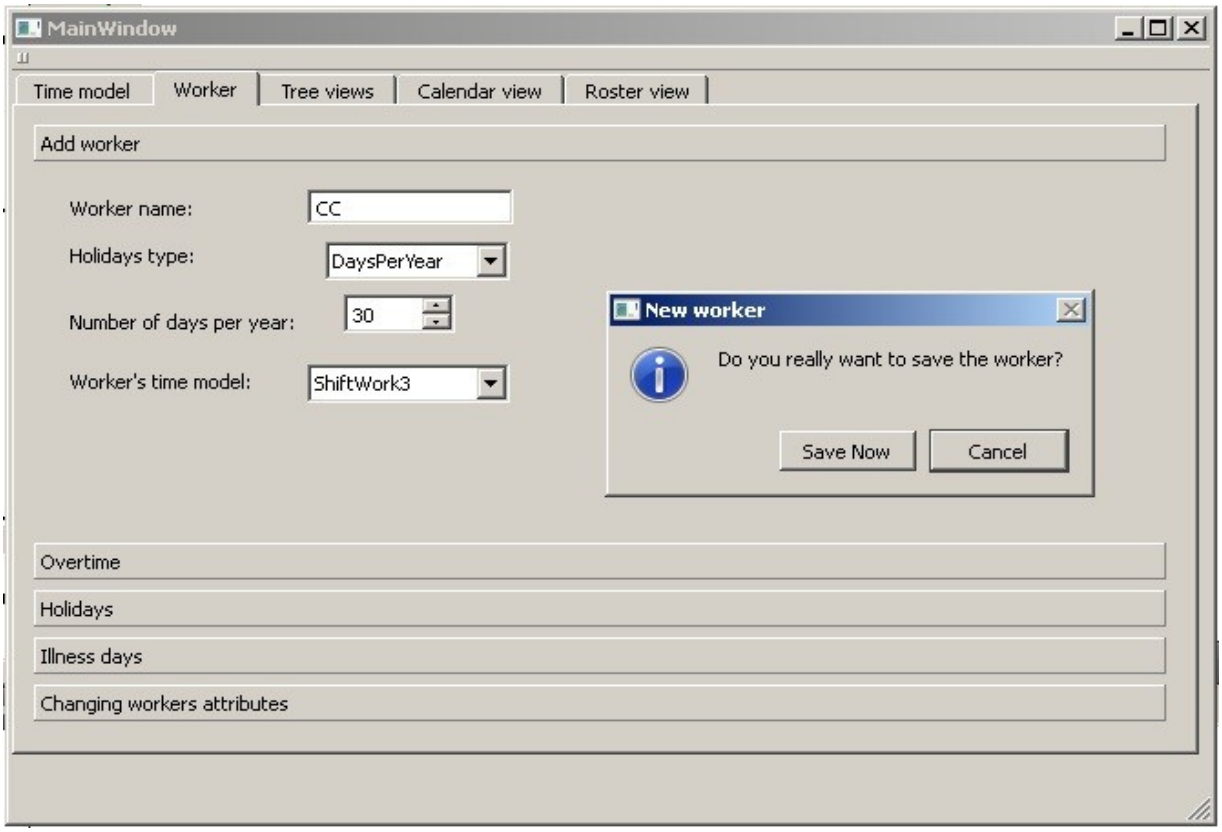


Illustration 30: Adding a worker

Following comes some output views of the program. On the picture below, the roster for the 3 workers builded up in this tutorial:

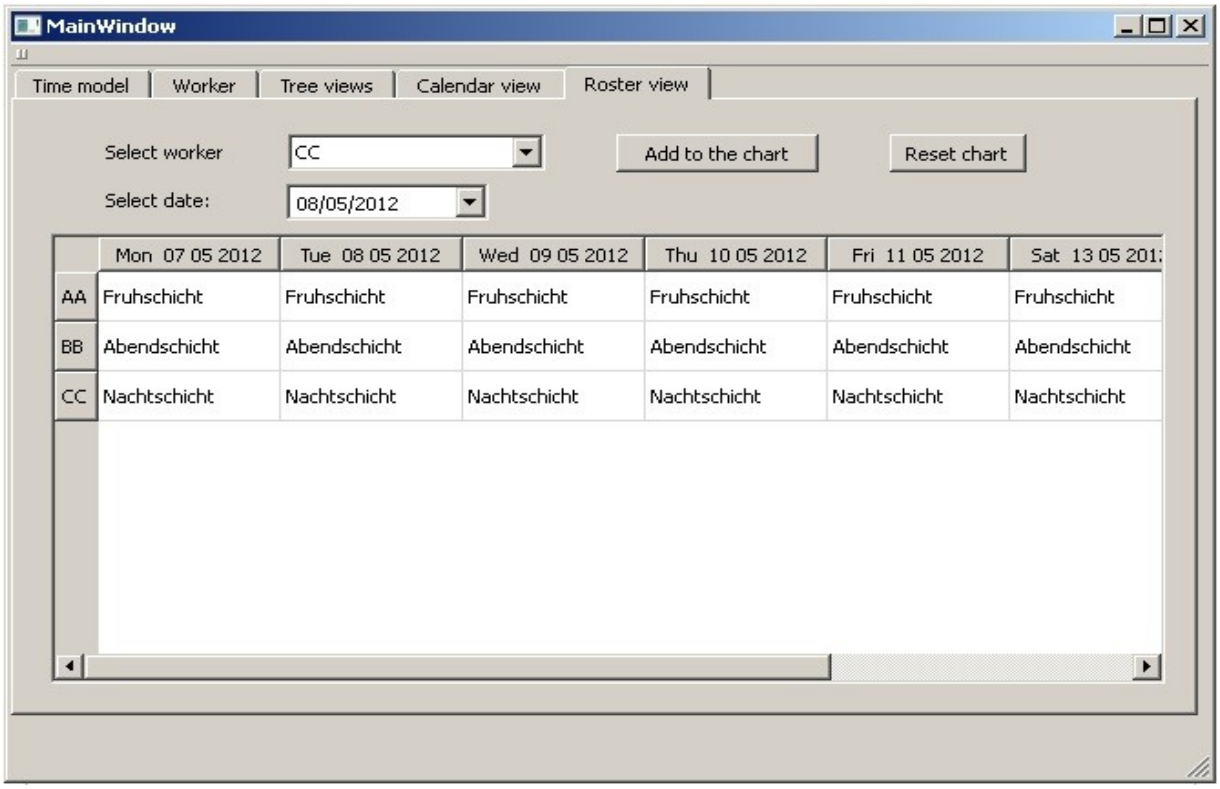


Illustration 31: Roster view of our 3-shift model with 3 workers

The calendar view lists the concrete ocupation (on holidays, on sickness leave, on sabbatical period, on workday, ...) for each worker on the system on the day selected by the user. Here is a screen-shot:

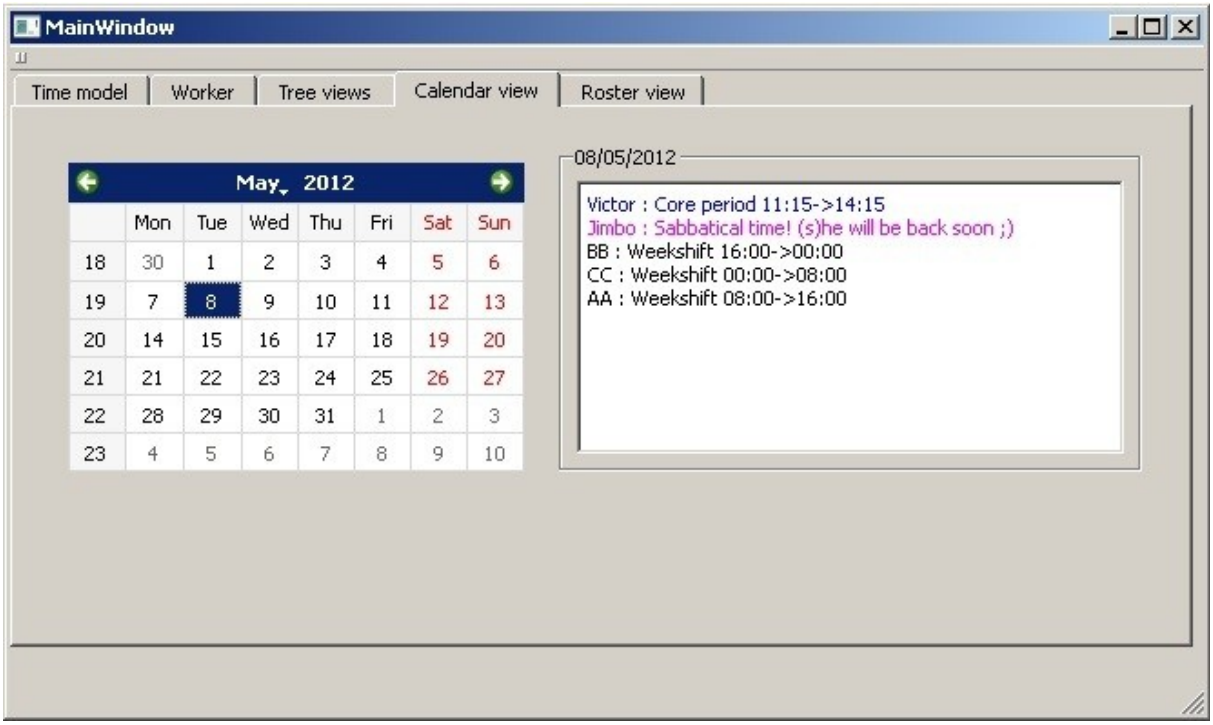


Illustration 32: Calendar view

And finally, the tree view of the workers and the time models in the system:

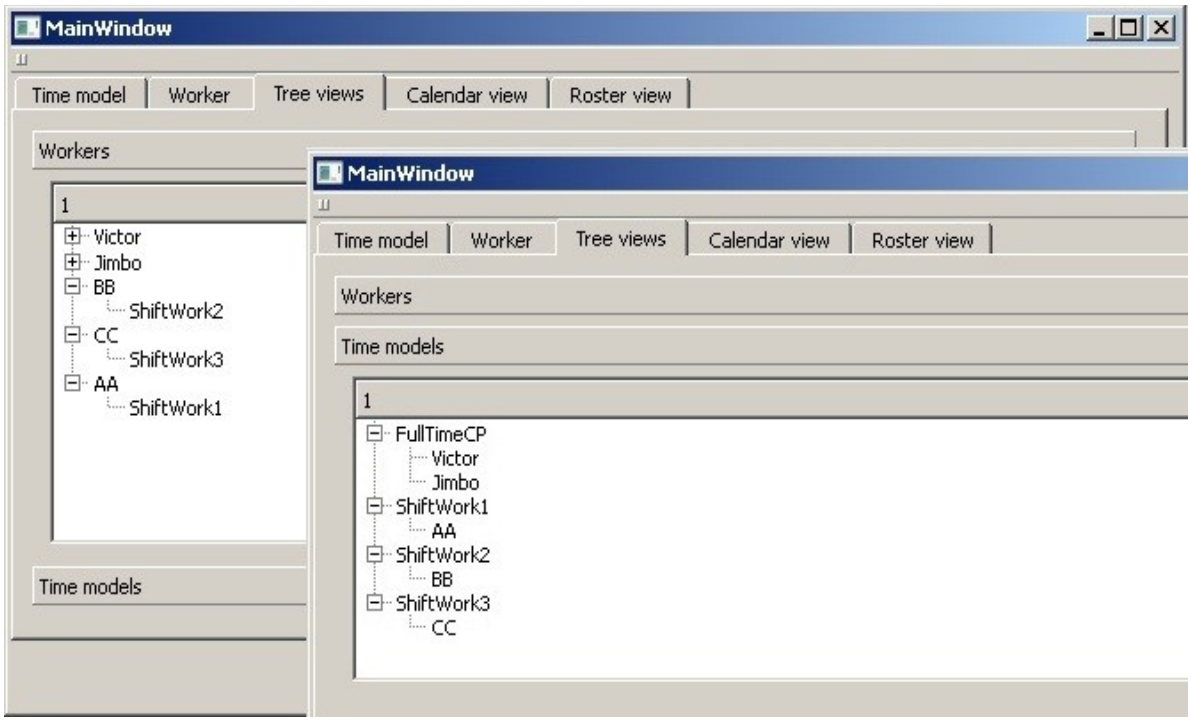


Illustration 33: Tree views of the workers (left) and the time models (right)

11. Appendix C: Survey

In this appendix the questions proposed to the people who completed the survey (extracted from STOWASSER, 2006, p. 419) are presented. In chapter 6, some references are made to this appendix. First, a list of the proposed questions is presented and then, a table with the results. The survey is presented in English. There is five possible answers: from 1 to 5 (1 is the worst answer and 5, the best).

Question 1: effectiveness

During working, can the program always fully and accurately fulfill project planning and controlling tasks and achieve the intended goals of users? Can it help you to complete your work effectively (regarding different tasks you deal with)?

(1) (2) (3) (4) (5)

Question 2: efficiency

With use of this program, can you save much time and effort to accomplish project planning and controlling tasks, in comparison with your experience before using it or with other similar software?

(1) (2) (3) (4) (5)

Question 3: task-oriented functionality

Does the program offer all the functions of project planning and control you expect?

(1) (2) (3) (4) (5)

Question 4: further task support functionalities

Can the program provide users with very useful project planning templates, meaningful default (alternatives), macros, etc. , which can further simplify the project planning tasks during execution in an acceptable way?

(1) (2) (3) (4) (5)

Question 5: use-oriented functionality

Is it simple enough for you to operate the program? How do you accept the handling with the program?

(1) (2) (3) (4) (5)

Question 6: quality of information

Do you find that the information that the program interfaces presents you are really helpful and task-related?

(1) (2) (3) (4) (5)

Question 7: quantity of information

Regarding your specific tasks, is the amount of information necessary and adequate for you?

(1) (2) (3) (4) (5)

Question 8: systems feedback

Does the program often provide some feedback, which provides useful explanation, hints or remind you of the possible consequence before carrying out the action?

(1) (2) (3) (4) (5)

Question 9: user requested information

How does the program react, if you request some specific explanation or information from it? Is it easy to find it and the results are always satisfactory?

(1) (2) (3) (4) (5)

Question 10: descriptiveness of information

Do you agree that the information given by the program is precise enough and its representation (terms, graph symbols, abbr.) is clear enough for you?

(1) (2) (3) (4) (5)

Question 11: controllability of information

Are you able to initiate a dialogue or control the direction and pace of it (e.g. go next/go back/finish/restart) until the goals can be really met?

(1) (2) (3) (4) (5)

Question 12: conformity of information

Do you find the design of interface and structure of the information consistent and well formulated?

(1) (2) (3) (4) (5)

Question 13: conformity with user expectations

Does the design of the program interface correspond to your knowledge and experience?

(1) (2) (3) (4) (5)

Question 14: input within the program

Can the program assist you to discover and avoid errors during operating it and automatically correct those evident minor mistakes, so as to get the same intended results?

(1) (2) (3) (4) (5)

Question 15: error handling

Are the error messages easily comprehensible and explain clearly where the mistakes are and how to easily correct them?

(1) (2) (3) (4) (5)

Question 16 suitability for individualization

Are you able to modify the forms of representation of the interface or the data according to your personal preference?

(1) (2) (3) (4) (5)

Question 17: suitability for learning

Does the program provide adequate help-information and variant learning possibilities (handbook, online-tutorial, hotlines) in order to help users to be familiar with the system?

(1) (2) (3) (4) (5)

Answers of the users:

Question	User 1	User 2	User 3	User 4	User 5
1	3	2	2	3	4
2	3	2	2	2	3
3	3	2	2	4	3
4	4	4	4	3	3
5	3	4	2	3	2
6	3	4	3	4	2
7	5	2	2	5	5
8	5	4	2	1	3
9	2	3	3	2	2
10	3	4	3	4	2
11	2	1	2	4	2
12	4	5	2	3	2
13	4	2	1	3	2
14	4	3	2	2	4
15	2	1	2	1	2
16	2	3	3	1	2
17	3	2	1	3	1

Table 2: Answers of the users to the questions of the survey

Answers scale over the different questions:

1: not at all

2: hardly, barely

3: medium

4: high

5: very high