Designing the User Interface of a Decision Support System for Power Production Planning

Design av användargränssnitt till beslutsstödssystem för planering av kraftproduktion

Master’s Thesis in Computer Science

2003-03-14

Johan Wilhelmson

Supervisor

Stefan Feltenmark,
Optimization Partner Sthlm AB

Examiner

Lars Kjelldahl,
Dept. of Numerical Analysis and Computer Science, KTH
Designing the User Interface of a Decision Support System for Power Production Planning

Abstract
SPOT (Seasonal Planning Optimization Tool), a decision support system based on a stochastic model for the seasonal planning problem in hydro-thermal power systems, is currently being developed, and it was the object of this Master’s project to design a user interface for it and to implement a prototype.

The development of the decision support system is part of the research project Optimization of Power Systems Under Uncertainty. Stochastic models are used to describe long-term planning problems in a more realistic fashion than deterministic ones. However, they add complexity when it comes to the visualization of the input and output data.

The report describes the user interface design process and the implementation of a prototype of the user interface in the programming language Java. It also describes ways to visualize a hydro-power system and its characteristic properties along with the solution to the optimization problem in the form of a scenario tree, along with general findings regarding the design of user interfaces for decision support systems based on stochastic models.
Design av användargränssnitt till beslutsstödssystem för planering av kraftproduktion

Sammanfattning
SPOT (Seasonal Planning Optimization Tool), ett beslutsstödssystem baserat på en stokastisk optimeringsmodell för säsongsplaneringsproblemet i hydro-termiska kraftsystem, utvecklas för närvarande. Målet för detta examensarbete var att utforma ett användargränssnitt för SPOT och att implementera en prototyp av det.

Utvecklingen av beslutsstödssystemet och dess matematiska modell är en del av forskningsprojektet Optimering av kraftsystem under osäkerhet. Stokastiska modeller används för att beskriva långtidsplanering på ett mer realistiskt sätt än deterministiska motsvarigheter. Dock är det mer komplex att visualisera in- och utdata.

Rapporten redogör för den process då användargränssnittet togs fram, och implementationsfasen då prototypen utvecklades i programmeringsspråket Java. Den beskriver hur ett hydrotermiskt kraftsystem och lösningar på optimeringsproblem som har formen av ett scenarioträd kan visualiseras, tillsammans med några allmänna slutsatser angående design av användargränssnitt till beslutsstödssystem som baserats på stokastiska modeller.
INTRODUCTION

1.1 The Company

1.2 The Assignment

1.3 This Report

BACKGROUND

2.1 Production In a Hydro-thermal Power System

2.1.1 The Hydro System

2.1.2 The Thermal System

2.2 The Power Market

2.2.1 The Physical Market

2.2.2 The Financial Market

2.3 Seasonal Power Production Planning

2.4 Optimization of Power Production Planning

2.4.1 The Decision Support System SPOT

2.5 The Problem

2.5.1 Design

2.5.2 Implementation

PLANNING THE PROJECT

3.1 Preparatory Studies

3.1.1 Visualization and Human Computer Interaction

3.1.2 Power Planning and Mathematical Optimization

3.2 Design

3.2.1 Requirements Gathering

3.2.2 Task Analysis

3.2.3 Envisioning Design

3.2.4 Evaluation of the Design

3.2.5 An Iterative Process

3.3 Implementation

3.3.1 Requirements of the Prototype

3.3.2 Platform for Implementation

3.3.3 Evaluating the Prototype

DESIGN

4.1 Requirements Gathering

4.1.1 Data Requirements

4.1.2 Functional and Usability Requirements

4.1.3 The Seasonal Planners’ Workflow

4.1.4 The Seasonal Planning Tool KR95
1 Introduction

This thesis represents the culmination of my education at the Royal Institute of Technology (KTH). Its purpose is to describe the master’s project that was conducted by me at Optimization Partner Stockholm AB during the summer and fall of 2002.

The project was supervised by Stefan Feltenmark, co-founder of the company and project leader of the research project on which I worked. Lars Kjelldahl at the Royal Institute of Technology’s Department of Numerical Analysis and Computer Science was my supervisor and examiner at KTH.

1.1 The Company

Optimization Partner is a consulting firm specializing in mathematical optimization problems. It was founded by Ulf Brännlund and Stefan Feltenmark, both PhD graduates of the Division of Optimization and Systems Theory at KTH. Optimization Partner’s expertise on modelling large-scale planning problems has led to the development and implementation of several decision support systems. During the year 2002, the company focused on power production planning using optimization techniques.

1.2 The Assignment

Power production planning involves making decisions about production, investment and trading. The time horizon of the planning stretches from operational day-to-day planning up to strategic planning for a period of several years. Seasonal production planning is of the latter type, typically with a time span of at least a year, so that the variations of the different seasons are incorporated.

Finding a suitable model of the seasonal planning problem is one of the goals of a research project called Optimization of Power Systems Under Uncertainty – a collaboration between Optimization Partner, the Division of Mathematical Statistics at Lund Institute of Technology and Fraunhofer Chalmers Research Centre for Industrial Mathematics.
1 Introduction

Due to the long planning horizon of the seasonal planning problem, the uncertainty of future events is one of the main features of the problem. After all, there is no way of knowing how much rain will fall the next autumn. This lack of information can be described using a stochastic model. The main advantage of this approach is that the problem is represented in a more realistic fashion than with a deterministic counterpart. However, this realism comes at a price: computation time. One way to make the stochastic optimization problem computationally tractable is to approximate the continuous data by discrete distributions. As this allows for a finite dimensional decision space, it results in a scenario tree, where each node represents an outcome, or scenario.

As a result of the above mentioned research project, a stochastic model of the seasonal planning has been defined (Feltenmark et al., 2000). Also, a decision support system based on this model has been implemented as a research prototype called SPOT (Seasonal Planning Optimization Tool). The development of an operational version of SPOT is currently one of the priorities of the project.

My assignment and thus the main goal of my master’s project was to design and implement a user interface for SPOT. The power planner needs to analyze and comprehend a large amount of data, and finding an interface design that aids the planner in making the best decision was a challenging task.

Visualizing the scenario tree in an intuitive fashion was to become an important part of the project, as the tree supposedly is a structure that most planners are not familiar with.

1.3 This Report

This report describes the process of designing and implementing a user interface for SPOT. It is my hope however, that some of the findings are general enough to be of value during the development of other decision support systems. The report is therefore primarily directed to the developer of such systems, although it is my hope that readers with a general interest in power production planning and/or software engineering will find a use for it. The comprehension of some terms
1 Introduction

specific to the domains of power production and the trading of power are needed to understand the details of the problem. These terms are generally explained the first time they appear.

To illustrate some of the design choices made, the report utilizes some of the diagrams defined by the Unified Modeling Language (UML). UML is an industry-standard language for specifying, visualizing, constructing and documenting the artifacts of software systems, and it was one of the tools used during the design phase of the project. However, no previous knowledge of UML is assumed of the reader, as these diagrams are included only for the sake of clarity.

Here follows a brief description of the different chapters in this report.

The next chapter, Chapter 2: Background, gives a more detailed background of the problem, including descriptions of power production in hydro-thermal power systems and the Nordic power exchange. It concludes with the definition of the problem, and the goals of this project.

In Chapter 3: Planning the Project, I describe the choice of methods, including the literature studies which began this project. This involves methods for designing the system, interviewing future users and implementing a prototype.

In Chapter 4: Design, the design of the user interface is discussed. The design phase also included user interviews and an evaluation, which is reflected in the text.

Chapter 5: Implementation describes the process of transforming the ideas and design of the interface to a runnable prototype.

Chapter 6: Discussion contains a general discussion where the results of the project, in the perspective of user interface design for decision support systems, are presented. Some recommendations on developing decision support systems based on stochastic optimization models are given.
1 Introduction

Chapter 7: Conclusions concludes the report, summarizing the results of the design and implementation phases.
2 Background

This chapter describes the general problem of optimizing power production in a hydro-thermal power system, starting with background information on power production planning and the power market. It is concluded by the definition of the problem of my master’s project and its goals.

2.1 Production In a Hydro-thermal Power System

A hydro-thermal power system is made up of a hydro system and a thermal system.

2.1.1 The Hydro System

The hydro system consists of river systems, i.e. hydro power stations and reservoirs connected by waterways (Figure 2.1).

![Figure 2.1. A schematic map of a simple river system.](image)

The water normally flows from higher altitudes to lower. In the hydro power stations the potential energy of the water is converted to electric energy. In reservoirs water is stored, either to control the flow through stations or to make up a reserve for dry years. Usually, small reservoirs in direct connection with stations serve the former purpose, while large sized reservoirs at high locations serve the latter.
2 Background

There are rules of both physical and jurisdictional nature which affect the operation of hydro stations and reservoirs. For example, reservoir level changes are regulated to avoid destroying the shores of the river stretch below.

The inflow to a river system shows a strong seasonal variation in the Nordic region, amongst other things due to snow melt. This means that there is very little inflow in winter followed by a big peak during spring flood.

These are some of the most important characteristics of a hydro system, which are significant when constructing an accurate model.

2.1.2 The Thermal System

The thermal system consists of independent thermal power stations. These are fossil fueled and nuclear power plants, where the operational cost is dominated by fuel costs (Feltenmark et al., 2000). There are typically also costs associated with starting and stopping the thermal power stations.

2.2 The Power Market

The largest part of the electricity is consumed by industry, service sector and households (Table 2.1).

<table>
<thead>
<tr>
<th>Demand</th>
<th>Consumer</th>
</tr>
</thead>
<tbody>
<tr>
<td>36%</td>
<td>Industry</td>
</tr>
<tr>
<td>29%</td>
<td>Service sector</td>
</tr>
<tr>
<td>26%</td>
<td>Households</td>
</tr>
<tr>
<td>7%</td>
<td>Losses and heat plants</td>
</tr>
<tr>
<td>2%</td>
<td>Agriculture</td>
</tr>
</tbody>
</table>


During winter, the consumption varies significantly with the outdoor temperature, since a major part of the households’ power consumption is a result of heating.
2 Background

The Nordic power market was deregulated during the 1990’s. In Sweden the network was separated from production with a state-owned company (Svenska Kraftnät) owning and operating the power grid. Any producer or consumer may connect to the grid for a fee.

The Nordic Power Exchange (Nord Pool) was established in 1993 and is currently the world’s only multinational exchange for trading electric power. Nord Pool offers a market for trading both physical and financial power contracts. Actors on this market include producers, distributors, industries and financial traders from the Nordic countries.

2.2.1 The Physical Market

Trade in power contracts for physical delivery is handled on Nord Pool’s spot market, also known as Elspot. At Elspot, power contracts are traded daily for physical delivery in the next 24-hour period. At 12 noon every day, participants submit their bids to make or take delivery on bidding forms covering all 24 hours of the next day. Typically, a power producer submits a bid to buy power if the spot price is low and sell power if the price is high. The spot price is then determined by the balance between bids and offers by all market participants (Figure 2.2).

![Figure 2.2. Determining the spot price.](image)

2.2.2 The Financial Market

Not only physical contracts are traded on the power market, but also purely financial contracts, including futures, forwards and options.
Futures and forwards are agreements to purchase electrical power at an agreed price on a specified future date. However, at that date there is no actual physical delivery, instead the transaction is settled financially.

An option is a right to buy or sell power at a fixed price before or on a future date. The purchaser of a buy option generally expects a strong spot price increase in the period leading up to the expiration of the option, while the issuer expects a stable and or gradually declining price development.

Producers trade these financial contracts in order to spread and handle their risks.

2.3 Seasonal Power Production Planning

A power producer controlling a hydro-thermal power system plans its power production, i.e. decides how to operate the hydro stations, reservoirs and thermal plants. This includes deciding on outflows from reservoirs, release in hydro power stations, thermal production, energy bought and sold on the spot market and contracts, futures, forwards and options positions.

The time horizon of this decision problem ranges from operational day-to-day planning to strategic planning that stretches over a period of several years. Seasonal planning is of the latter type, typically with a time span of over one year, to allow the planner to take the seasonal periodicity of the inflows and power consumption into account.

The common goal of all types of production planning is usually to maximize the producer’s profits. However, maximizing the profits over the planning horizon would be short-sighted. Most companies probably want to continue doing business for a longer time than the forthcoming planning period. Emptying all reservoirs at the end of the year would give a short-term income, but could be disastrous for the next year. Therefore, one must also consider the value of the water stored in the reservoirs at the end of the period.
2 Background

A great difficulty that arises when increasing the planning horizon is foreseeing future values of variables. For example, there is no way of telling how much rain will fall next autumn, or what the spot price will be a year from now. This lack of information can be handled by a stochastic model of the planning problem.

2.4 Optimization of Power Production Planning

A research project, led by Stefan Feltenmark at Optimization Partner, has defined a stochastic model of the seasonal planning problem. This means that variables such as inflows to stations, spot price etc. are continuous stochastic variables. The distributions of some of these variables can be estimated to a reasonable degree of accuracy using historical data.

In the deterministic case, the objective function represents the total profit earned over the planning period plus the value of the water left in reservoirs at the end of the period. However, since some variables are stochastic, this function would also be a stochastic variable. To obtain an optimization problem, the objective function is the expected utility of this stochastic variable. The choice of utility function reflects the risk aversion of the power producer, and must therefore be selected with care.

The stochastic model is more realistic than its deterministic counterpart. However, the realism comes at a cost: computation time. To make the problem computationally tractable, you approximate the continuous distributions with discrete ones.

The solution to a stochastic optimization problem with discrete distributions comes in the form of a scenario tree, where each node represents a collection of outcomes of the stochastic variables, along with the optimal values of the decision variables. To every node, there is an associated probability of a given scenario containing that node (Figure 2.3).
2 Background

Figure 2.3. An example of a scenario tree with probabilities marked in each node.

2.4.1 The Decision Support System SPOT
The development of a decision support system called SPOT (Seasonal Planning Optimization Tool), based on the stochastic model of the planning problem, has been initiated. SPOT will allow a seasonal power planner to set up and solve a problem according to the stochastic model, and to evaluate the solution of the problem. The system has previously been implemented as a research prototype in Matlab, and is currently being rewritten from scratch in ANSI C++.

2.5 The Problem
The goal of my master’s project was to:

- Design a graphical user interface for SPOT.
- Implement a prototype of the user interface.

The typical user is a seasonal planner at a power production company, who is assumed to have expert knowledge in power production planning, and little or no experience in mathematical optimization.
2 Background

A user group consisting of two seasonal planners at the Nordic energy company Fortum had been formed. These users were accustomed to the deterministic planning tool KR95, which is further discussed and contrasted with SPOT in Section 4.1.4.

2.5.1 Design

The goal of the design phase would be to construct a design of the graphical user interface, which would allow the users to work with SPOT in an efficient manner. This includes functionality for setting up a problem, solving it and evaluating the solution. As the solution is a scenario tree, a data type assumed to be unfamiliar to the user, it would be important to concentrate on finding an appropriate way to visualize it.

2.5.2 Implementation

The goal of the implementation would be to realize a prototype of the user interface runnable on a personal computer. This prototype would be consistent with the design where possible, and its source code would form the basis of the final product.
3 Planning the Project

The project was logically divided into three separate stages: the preparatory studies, followed by the design of the user interface and finally the implementation of the prototype. In this chapter I describe the methods I chose to use in the different stages.

3.1 Preparatory Studies

Since the project would focus on the user interface of SPOT, the preparatory studies included studying books and articles on visualization and human computer interaction, as well as literature on power planning and mathematical optimization. In the following section I briefly discuss my personal opinions on these books and in what way they were helpful in my work.

3.1.1 Visualization and Human Computer Interaction

*The Visual Display of Quantitative Information* (Tufte, 2001) is somewhat of a bible in its domain, although it does not consider computer applications or other interactive media. Tufte’s book acted as an inspirational guide through the design stage. Tufte teaches a graphical economy where each stroke of the pen should have some information value. The discussion on the value of a clear and precise visual display is so general that it is straightforward to apply it on computer visual design.

*Human-Computer Interaction* (Preece et al., 1994) and *Information Visualization* (Ware, 2000) complemented Tufte’s book by discussing visualization and interaction using computers specifically. The former presents an overall view of human-computer interaction, including development models and prototyping techniques, while the latter describes different visualization techniques in depth. My workflow during the design stage was to be inspired by Preece, while Ware offered an alternative perspective to Tufte, with its more modern approach.
3 Planning the Project

3.1.2 Power Planning and Mathematical Optimization

I also studied power planning, the Nordic power market and the mathematical model that SPOT is based on. This included lectures given by Ulf Brännlund and Stefan Feltenmark at Optimization Partner on mathematical optimization in general, specifically linear and stochastic programming.

Nord Pool’s web site¹ was the primary source of information about the Nordic power market. I have also had the opportunity to frequently meet the rest of the researchers involved in developing SPOT to discuss the details of the model.

3.2 Design

One could separate SPOT into two main parts: the kernel, which receives problems as input and outputs solutions to these problems, and the user interface, which lets the user define problems and visualizes their solutions in a suitable way.

The first of the project’s two main objectives was producing a design of the user interface. Before I would be able to envision a suitable design, I found that three questions had to be answered:

- What input does the kernel expect, and what output does it produce?
- What goals do the users expect to achieve when using SPOT?
- What tasks do the users perform to reach these goals?

The answers to the first two questions define the requirements of the system. The process of finding them is thus called requirements gathering. The third question will then be answered using task analysis.

¹ See http://www.nordpool.no.
3.2.1 Requirements Gathering

Requirements gathering means finding the requirements on the system from the user’s point of view. This includes data requirements (the input to and output from the kernel), functional requirements and usability requirements.

The data requirements were essentially determined by the model on which SPOT was based, while the identification of the functional and usability requirements would be approached differently. A SPOT user group had been formed, consisting of two power analysts at the division of seasonal planning at Fortum. By interviewing them, I was hoping to identify these types of requirements.

Interviewing the users is a general method often used in requirements gathering. In an interview it is easier to become aware of the users’ subjective opinions, which traditionally have a strong influence on the development of a system. I was also hoping to find possible problems with the decision support system currently at use by the planners, to avoid repeating those mistakes myself.

I had to decide on what interviewing style to use. I found that the style most appropriate for this interview was the semi-structured interview (Preece et al., 1994) with some set topics and questions, allowing me as the interviewer to follow the interviewee’s replies. It contrasts the structured interview, typically found in public opinion surveys, with a set of fixed questions and little or no flexibility. The semi-structured interview tends to be less formal.

I also decided on interviewing both users together. I thought that it would allow them to reflect on and be inspired by each others’ answers, and that it would be constructive. I prepared a rough interview plan on paper (see Chapter 10: Appendix) so as not to forget any important questions.

3.2.2 Task Analysis

The most important goals when using SPOT should be identified by this stage. By analysing and structuring the tasks performed to reach
3 Planning the Project

these goals I wanted to obtain the knowledge necessary to develop a design.

The concept of task analysis treats the different tasks a user will want to perform in order to achieve their goal. One usually makes a distinction between actions and tasks. Actions are simple activities that require little or no thought, like moving the mouse or pressing a key, while tasks are more abstract, like entering text or saving a document. Tasks are performed in order to achieve goals.

There are a number of different techniques to use in task analysis, where hierarchical task analysis is one of the most well known. It is an iterative process for categorizing and identifying tasks by breaking them down into subtasks. The result of the hierarchical task analysis is typically a structure chart where a goal is at the top and the tasks and subtasks that are performed to reach the goal is found underneath it. I decided to use this approach, mainly due to the fact that it has been shown to be a sound and flexible method.

3.2.3 Envisioning Design

The results from the requirements gathering and the task analysis would now lead to the design of the user interface. This step involves making decisions on the grounds of the task analysis. There is however not one precise way to make these decisions, and it is here the craft of user interface engineering is most prominent.

Still, there are several tried-out methods like the structured approach of the Command Language Grammar (Moran, 1981), or holistic approaches like sketching techniques and storyboards.

I decided that prototyping the user interface using mock-ups on paper would best suit my work process. The mock-ups describe the physical details of the design, while still capturing the general concept. This would also allow me to design the user interface without deciding on the platform of the implementation at this stage in the project. The mock-ups could without difficulty be incorporated in the evaluation with the users.
3.2.4 Evaluation of the Design
Without an evaluation of the user interface design, it is difficult to know whether or not the design fulfills the needs of the users. An evaluation can be just getting informal feedback on the design, or it could be a rigorously planned laboratory experiment or large-scale survey. Regardless of the type of evaluation, an important goal of it is to find potential problems before they are incorporated in the system and are much more difficult to fix.

By discussing the paper prototypes with the users, I hoped that they would get a clear image of what my design would look like without hesitating to suggest changes or criticize it.

3.2.5 An Iterative Process
Usually, the design process is iterative, requiring the developer to rethink the design after the evaluation, implement any changes suggested and perform yet another evaluation. Due to the short time span of this project, I realized that there probably would only be enough time to perform one whole iteration.

However, for the evaluation to have any significance in the scope of this project, the results from it would be used to make some final adjustments on the design.

3.3 Implementation
The second goal of the project was to implement a prototype of the user interface design. I needed to decide on what tools to use when transforming the interface design from paper mock-ups to a functioning prototype.

3.3.1 Requirements of the Prototype
During an initial meeting with the SPOT research team, it was decided that a high level separation of user interface and data model was desirable, due to the following reasons:
3 Planning the Project

- The data model would be developed simultaneously and independently from the user interface.
- It would be an advantage if the kernel could be run on a host separate from the user interface. While the kernel may be parallelized and run on a multiprocessor computer, the user interface should be runnable on a personal computer, communicating with the data model via an Internet connection.
- Due to the fact that both Unix and Windows platforms were being used by the research team, it would be preferrable if SPOT would run on both these platforms.

3.3.2 Platform for Implementation

While platform independency of the data model would be enforced by implementing it in the standardized programming language ANSI C++, using that language for a user interface was considered unsuitable, since the language itself does not provide any support for graphical applications.

The object-oriented programming language Java\(^1\) was found more appropriate for the implementation of the user interface. One of the advantages of Java is its platform independency, due to the fact that Java programs run on a virtual machine. The virtual machine translates Java instructions to actual machine code. There are Java virtual machines for most common operating systems, including Windows, Unix, Linux, MacOS. The virtual machine also supports graphical applications that behave in the same way on all platforms.

The connection between the Java user interface and the C++ data model would be realized with CORBA\(^2\). CORBA is a high-level architecture

\(^1\) For further information about the programming language Java, see http://java.sun.com.

3 Planning the Project

that computer applications use to work together over networks, regardless of what programming languages they were implemented in.

A modelling language is to the software developer what the blueprint is to the building constructor. I decided to use the industry-standard Unified Modeling Language, UML, to specify, construct and document the user interface on the implementation level.

3.3.3 Evaluating the Prototype

Finally, I wanted to conclude the project by evaluating the prototype along with the SPOT user group. An evaluation would indicate if the design had been implemented successfully.

Due to the fact that relatively few users formed the user group, I decided on performing the evaluation of the prototype in a similar fashion as the earlier evaluation of the design, that is as an informal interview.
4 Design

This chapter describes the findings of the requirements gathering and the task analysis, and is concluded by the resulting design of the SPOT user interface.

4.1 Requirements Gathering

While the data requirements of SPOT were determined by the model on which it was based, the functional and usability requirements were established by interviewing the user group. Since the model is described in detail by Feltenmark et al. (2000), I will give a brief overview of its main features.

4.1.1 Data Requirements

One could separate SPOT in two main parts: a computational kernel and a user interface. The data requirements are defined by the kernel’s input and output, in other words: what data is needed to describe problems and solutions.

The Problem

A problem consists of a hydro-thermal power system, a contract portfolio and a few additional parameters. The hydro power system is defined by its components and the waterways that connect them. Variables related to the components in the power system are for example:

- The name of the component.
- Limits such as $\bar{x}^h_{i,t}$ and $\underline{x}^h_{i,t}$: the upper and lower limits on release from hydro power station $i$ during time period $t$.
- $g^h_i$, a function describing the input-output relation (inflow to generation) for hydro power station $i$.
- $g^m_i$, a similar function describing the fuel cost in thermal plant $i$.

The computational kernel will need a utility function that basically describes the risk-averseness of the power producer. It will also need
historical data like inflows to be able to produce a solution. The historical data will however be automatically collected from a database. It is complemented by observations of last week’s reservoir levels, inflows etc. in order to generate scenarios for the solution.

The Solution
The kernel calculates a solution iteratively using linear programming techniques. This means that all functions must be approximated by piecewise linear functions. For the problem to be solvable, they must also be convex, or concave in the case of fuel costs for thermal power generation. Thanks to the method used\(^1\) when solving the problem, an upper and lower bound on the optimal solution is produced. This can be used as a measurement of the accuracy of the current solution.

The solution to a problem is represented by a tree structure, where every node \( n \) has exactly one ancestor node \( a_n \), except the root node of the tree, which has none. The levels of the tree correspond to time periods. A path from the root node to another node is called a scenario.

To every node \( n \) in the tree, there is an associated transitional probability, \( \tau_n \), i.e. the probability that a given scenario will contain the node \( n \) if it contains its ancestor \( a_n \). There is also the probability \( \pi_n \), i.e. the probability that a given scenario contains the node \( n \), which is given by:

\[
\pi_n = \pi_{a_n} \cdot \tau_n
\]

Also, associated with every node is a vector containing values of the model’s variables, for example \( x_i^h \), the release from hydro station \( i \), and \( g_i^h \), the generation in power plant \( i \).

\(^1\) Nested Bender’s decomposition.
4 Design

4.1.2 Functional and Usability Requirements
The functional and usability requirements were to be identified mainly by interviewing Anna Höök and Johan Andersson, two of the four power analysts at Fortum’s division of seasonal planning. The interview was conducted according to the plan and started with their descriptions of the typical workflow. There was a discussion of the planning tool currently in use: KR95. The users were queried on their thoughts on the user interface of SPOT in contrast with KR95. I was also treated to a demonstration of the most common tasks when using KR95.

4.1.3 The Seasonal Planners’ Workflow
The assignment of the seasonal planning division is to produce a plan for the seasonal production in Fortum’s hydro power system. This plan is updated every week, and is merged with short-term and financial plans, to form the next week’s operational plan. The operational plan includes, amongst other things, upper and lower bounds on the spot price for production at different hydro power stations.

Every Wednesday the seasonal planners meet with representatives of the other divisions: operational planning, financial analysis etc. The goal of this meeting is to reach a mutual agreement on the next week’s operational planning. On the Tuesday before there is a preparatory meeting where no decisions are made, but where the different divisions express their wants and needs.

It is mainly before these two meetings that a planning tool, KR95, is used by the seasonal planners. Reports on the inflows and reservoir levels of the past week are received on Monday afternoon. This data, along with a spot price prognosis from the division of financial analysis, weather forecasts etc. is entered into KR95, which in return delivers a plan for next week’s operations. Reports on this plan in the form of time-series of inflows, release etc. of some strategic and representative hydro power stations are printed.

On the Tuesday meeting, the printed time-series are discussed. The different divisions present their views on the plan. Typically, the plan cannot be implemented in its entirety, and has to be adjusted. For
example, the financial analysts may now have indications on a somewhat higher spot price, which will call for a higher production.

After the meeting on Tuesday, the suggested adjustments are made in KR95. A new plan is produced, which is typically agreed upon on the Wednesday meeting.

At the end of the week, the seasonal planners produce plans for 41 different inflow scenarios. These plans are used by the division of financial analysis for portfolio management.

4.1.4 The Seasonal Planning Tool KR95

The different divisions all have their own planning tools. The seasonal planners exclusively use KR95. It utilizes a linear network optimization model to produce a deterministic seasonal plan. It is integrated with the spreadsheet software Microsoft Excel 97. Four analysts take turns in using it for about one week each, though Anna Höök had used it more frequently than the others.

Two main problems with KR95 were identified:

- The solutions produced by KR95 would sometimes be extreme (certain variables would either be at their maximum or minimum values, never in between), and were in these cases deemed as unrealistic. This weakness had been discussed in an evaluation performed by Optimization Partner¹, and identified as shortcomings of the linear network model utilized by KR95.
- Producing the 41 plans for the financial analysis division had not been automated, thus requiring the user to repeat a monotonous task, i.e. change a setting and press a button, 41 times every two to three minutes.

¹ Optimization Partner (2001).
4 Design

KR95 has no support for financial trading in contracts, futures, forwards etc. The analysts, however, saw no need for this support, since this was the responsibility of the financial analysis division, and it had its own tools.

The analysts thought that it was positive that KR95 was integrated with Excel, since Excel’s high degree of flexibility allowed them to create and modify their own reports and write macros to automate the data import into the system. For example, reports on inflows and reservoir levels would often be received in the form of Excel documents, and there were macros for automatically extracting the data from these documents into KR95.

The KR95 user interface consisted of an Excel spreadsheet with a relatively large number of sheets. The stations and reservoirs of the hydro power system were listed by name in long lists, but a search control had been provided, letting the user type the first few letters of a name to find it in the list. The input parameters could be entered manually, but there were macros that automatically extracted this data from other Excel files. When producing a plan, a process that took around two to three minutes, KR95 offered no feedback whatsoever, until the plan was ready, or it could not be calculated, which prompted an error dialog.

4.1.5 The SPOT User Interface

Functionality

The users expressed a wish for a higher degree of control over the calculation than KR95 offered. Setting limits on the accuracy of the solution, the maximum time, number of iterations, and aborting a calculation were suggested tasks when producing a plan.

They also said that if SPOT was to be used in the way KR95 was being used now, the time to produce a solution should preferably not exceed five minutes. However, if the solving required little supervision by them, they thought that the time needed was of lesser importance.
4 Design

Since a large amount of data was available only as part of Excel documents, exporting and importing, or cutting and pasting, to and from these documents were functions that were wanted.

Visualization
The users thought that being able to view power system components in some sort of map would be of value. A list of the components would only be of value if it could be ordered according to different criteria, since Fortum’s power system included over 80 components.

Some of the properties of components in the power system could be indicated using several different units, for example the release of water in a reservoir which could be stated as m³ or as the percentage of maximum release. The users believed that being able to select the appropriate unit was beneficial.

Regarding the scenario tree, they wished for some way of displaying “the most probable way through the tree”, as one of them expressed it. It was my understanding that they would prefer it if the scenario tree was plotted as a regular two-dimensional time-series (Figure 4.1), rather than for example a three-dimensional representation (Figure 4.2). It was not necessary to be able to view and compare data of for example different hydro plants at the same time.
4 Design

Figure 4.1. A two-dimensional scenario tree representation where probabilities are not visualized.

Figure 4.2. Example of a three-dimensional scenario tree representation, where each level in the tree is plotted as a histogram where the height of a column indicates probability.

4.2 Task Analysis

Due to the fact that a solution produced by SPOT has the form of a scenario tree, it would not be necessary to run the application several times for several scenarios. Furthermore, the SPOT model would not create jerky solutions, as was the case with the current tool, KR95\(^1\).

\(^1\) Optimization Partner (2001).
4  Design

Thus, the two main problems that the planners experienced when using KR95 would be resolved by SPOT.

The planners seemed largely satisfied with the workflow of KR95, where one first defines a problem by inputting and importing the data that defines it, then create paper print-outs on the solution. I saw no reason for abandoning this logical chain of tasks, thus the main goal and the tasks performed to reach this goal was found to be:

0. Solve problem and reveal solution (goal).
   1. Enter SPOT.
   2. Define problem.
   4. Evaluate solution.
   5. Print reports.
   6. Leave SPOT.

By analysing each of the six tasks, they were broken down into subtasks, which in turn were broken down. I will give some examples of this process, known as hierarchical task analysis.

**Task 0.2: Define Problem**

A problem consists of a system, a contract portfolio and some additional parameters. To define a problem, one should define each of these parts. However, if SPOT is run once a week, it is likely that the problem definition to a large extent will be equal to that of last week. For example, only small changes will have occurred to the power system. Therefore it is important to be able to restore the problem of last week, and to work from that.

Thus, the subtasks when defining a problem are:

1. Restore problem.
2. Modify system.
3. Modify contract portfolio.
4. Modify additional parameters.
Task 0.3: Solve Problem
The actual solving is not performed by the user per se, the user only has to send the problem to the computational kernel, for example by pushing a button. However, since the calculation most likely will take in the region of a couple of minutes, feedback must be provided to the user. This includes:

- Could the kernel be contacted?
- Did it receive the problem definition properly?
- How much time has passed since the calculation started?
- How accurate is the current solution?

It is also important that the user is able to abort the calculation and return to defining the problem, perhaps if they realized that the problem definition was not correct. It would also be valuable for the user to abort the calculation, but receive a partial solution if there is one, perhaps if they see that the accuracy of the solution is adequate.

4.2.1 Result of Task Analysis
Breaking down the tasks and subtasks mentioned above iteratively in this fashion eventually resulted in the hierarchical task analysis presented in Figure 4.3.
Figure 4.3. The results of the hierarchical task analysis.
4 Design

4.3 Envisioning the Design
The SPOT user interface would now be envisioned by interpreting the results from the requirements gathering and the task analysis.

As it was indicated by the user group that they found the financial aspects of SPOT (i.e. the contract portfolio and risk management) less important, I decided to focus on the physical aspect (i.e. the hydrothermal power system).

4.3.1 Main Views
As mentioned above, one could divide SPOT into two logical parts: the user interface and the kernel. The kernel can be thought of as a black box, whose interior is unknown and irrelevant to the user. What matters is its function: it receives as input the parameters that define a problem, and outputs the solution to the problem (Figure 4.4).

This black box metaphor led to splitting the SPOT user interface into two main views: Problem and Solution. The user defines the problem in the former view, and then switches to the latter view to receive and evaluate the solution.

4.3.2 Problem View
In the Problem view the user sets up the problem, that is specifies all the input to the black box, according to the metaphor. This includes defining the power system, the contract portfolio and the additional parameters.
Restoring a Problem
The first task when defining the problem was restoring an earlier one. The user will most probably want the last problem defined, so that could be restored automatically. However, there are situations where it would be desirable for the user to restore for example the third last problem. Therefore, providing a list of all or at least a number of the last problems defined, sorted by date in descending order, would seem natural. The user could then select another problem to restore it.

When a problem is restored, the user modifies the system. Modifying the system is the same as modifying the components of the system (see Chapter 4.2.1). Modifying components are performed by selecting them, and then modifying their parameters.

Selecting Components In the System
It is easier to recognize material than recall it from memory. Based on this, a clickable map of the system, geographic or schematic, could be a more effective way for the user to select components in a power system than typing the name of the component, or selecting it from a list.

The geographic map offers a natural way of finding the location of a component. However, it will probably be necessary to zoom in to find components that are so close to each other that they overlap, and zoom out to find components that are too far away from each other to both show in the visible area. A schematic map on the other hand, guarantees an economic way of displaying many components without them overlapping each other (see Figure 4.5).
4 Design

Since the thermal plants are disconnected, it is difficult to display them in a natural way on a schematic map. Despite this, it could be useful to let the user arrange the plants, for instance like the desktop icon metaphor.

I decided to implement both geographic and schematic maps, and let the user choose which to employ. These maps would allow the user to select several components at the same time to modify the parameters they have in common.

Modifying the Properties of Components

Some of the properties are time-dependent functions, like the bounds on release from a reservoir. Visualizing these by plotting them as time-series offers a fast way to see if they are reasonable, but defining them would be done more effectively using a table (Figure 4.6).

There are also piecewise linear functions such as the fuel cost function of thermal plants. The fuel cost function is required to be concave, and if it is not, it could be automatically approximated with a legal piecewise linear function. These types of functions are generally defined once, and rarely modified.
Switching To the Solution View

When the problem has been defined, the solution has to be calculated, i.e. the input has to be sent to the black box, which after a while will send the output back. Thus, when the user switches to the Solution view, the problem is automatically sent to the kernel.

4.3.3 Solution View

Reporting the Progress of the Calculation

While the solution is being calculated, the Solution view should display a progress report. This way the user will see if the accuracy of the current solution is already satisfactory, or if the solution does not converge and the calculation should be aborted.

As mentioned above, the computational kernel is able to provide an upper and a lower bound on the optimal solution after each iteration. The difference between these bounds, the gap can be used as a measurement of accuracy, and be plotted as a time series with the corresponding index of the iteration on the horizontal axis (Figure 4.7). This shows not only the current accuracy, but also the development of the solving process.
The user should be able to abort the calculation and return to the Problem view by pressing a button, with a confirmation dialog box in case the button was pressed by mistake. It should also be possible to abort the calculation, and fetch the solution produced in the last iteration, also by pressing a button with an additional confirmation dialog box.

If the calculation is not aborted, it continues until the time limit, the maximum number of iterations, or the requested accuracy has been reached, and the Solution view is displayed.

When a solution has been produced, the user will want to select and view a property of a component. When setting up the problem one would also select components using geographic and schematic maps (see above). These maps should be used in the Problem view too for consistency.

**Viewing the Scenario Tree**
With a component and a property selected, the corresponding values in the scenario tree can be plotted. As was discussed above, a two-dimensional representation was favoured by the users over a three-dimensional variant.
4 Design

As the tree branches, discerning between different scenarios will be difficult. Since the number of scenarios typically exceeds 100 even in trivial problems, colour coding them is problematical\footnote{Horton (1995) claims that about 50 colour codes is the maximum number a viewer with extensive training can recognize. The optimal number is expected to be substantially less.}. Due to the relatively low resolution of a computer monitor, the tree would become a chaotic blend of different colours after just a few levels.

If one chooses not to colour code the different scenarios, could one then use colour as a mean for visualizing the different transitional probabilities? For example, the higher intensity of red, the closer the probability is to 1. However, the same problem would arise as the tree branches, making this approach ineffective.

Drawing edges with a thickness relative to the probability would be straightforward and fairly intuitive, but it would only work for the first few levels in the tree, since the differences between the probabilities decrease as the tree branches, and trying to visually exaggerate the difference within one level would cause problems between different levels.

I decided on implementing an interactive tool to select levels in the tree, thereby allowing a visualization of the probability distribution associated with the nodes at that level, for example by utilizing a histogram (Figure 4.8). However, when using a histogram, care must be taken when selecting the number of bins. Features of the data are hidden if there are too few or too many bins. As the appropriate number varies with the data, a control for setting the number of bins should be easily accessible, perhaps even in direct connection to the histogram.
This interactive tool is also used for discerning between scenarios. By selecting a node, for example by clicking on it using the mouse, the scenario associated with the node could be highlighted (Figure 4.9).

When selecting a node, values that are associated with the selected component could be shown, offering a more detailed view than the rather rough view of the whole tree. If the subtree of the node branches, the scenarios could all be highlighted showing the different possible outcomes.

4.3.4 Results of Envisioning the Design
Simple mock-ups of the user interface were created. The objective of these paper prototypes were not to describe the layout of the interface
4 Design

in detail, but rather to give an impression of the overall structure and functionality of it.

The Problem view (Figure 4.10) and Solution View (Figure 4.11) are depicted below.

![Figure 4.10. The Problem view. To the left, the available problems are listed, while the properties of a selected station is presented to the right.](image)

![Figure 4.11. The Solution view, illustrating the scenario tree visualization.](image)
4.4 Evaluating the Design

The paper mock-ups of the user interface were presented to the two seasonal planners at Fortum. The overall reaction to the design was positive. Below are the most important opinions that were voiced during the evaluation.

The Power System

The planners thought that the idea of selecting components from a map over the system was good. However, they stressed that it would be valuable if it was complemented with a text field for typing the first few letters of a component to select it. This fast selection mechanism was something KR95 provided, and which they appreciated.

Progress Report

To be able to specify the maximum time for the solution of a problem, and to be able to abort or skip ahead in the middle of a calculation, were functions that were appreciated. Also, displaying feedback during the calculation of the solution was something the users missed in KR95, where essentially nothing indicated that a solution was being calculated.

Visualizing the Solution

Regarding the interactive histogram distribution plot, the users were hesitant. They seemed to prefer visualizing the distribution in the time-series view, instead of using an interactive tool. I suggested using the percentiles and/or mean to roughly visualize the probability distribution, which they agreed would be satisfactory.

The main change to the design that was suggested by the evaluation was allowing the user to select a component by typing its first few letters.

4.5 Results of the Design Phase

I considered the design phase successful. The users seemed positive judging by the evaluation. Hence, I decided that after implementing the
4  Design

few changes suggested in the evaluation, the design was satisfactory
and could form the basis of the user interface of SPOT.
5 Implementation

Due to the requirement of a high-level separation of the SPOT user interface and the computational kernel, it was decided that a client-server model would be ideal for the entire system. The server (or the black box, according to the metaphor from Section 4.3) consists of the computational kernel that communicates with a database and an external linear-programming solver (Figure 5.1).

![Figure 5.1. A conceptual view of the complete SPOT system.](image)

The kernel was to be implemented in ANSI C++ for speed and platform-independency, while the user interface was written in Java for cross-platform functionality and graphics support. The interaction between the user interface and the server was handled by CORBA. This solution meant that one could choose to run the client and server either on the same machine, or on two separate ones. In the latter case, one could use a basic desktop PC for the user interface, and a dedicated server to shorten computing times.

Since the kernel was being implemented in parallel and did not reach a stage of being able to communicate with the interface during the
5 Implementation

project, simple power systems and scenario trees were made up when testing and demonstrating the prototype.

5.1 Structure
Java is an object-oriented programming language that encourages code reuse. Thanks to the vast class library distributed with Java, several classes used in the user interface could be derived from specialized standard components.

Central to the user interface implementation is a document (essentially the model according to the Model-View-Control paradigm\(^1\)). This consists of the problem and a solution if any has been calculated. The document is displayed through views and manipulated through controls (Figure 5.2).

![Figure 5.2. Overall view of the internal structure of the user interface.](image)

The document listens to changes in the problem and the solution according to the Observer design pattern\(^2\), and sends messages about these changes on to the observing user interface, which is then updated.

---

\(^1\) See http://ootips.org.

\(^2\) Ibid.
5 Implementation

To illustrate the implementation on a more detailed level, two of the main structures of the problem and solution: the hydro-thermal power system and the scenario tree, are briefly described in the next subsections.

5.1.1 The Power System

The power system, which is a defining part of the problem, is derived from two abstract base classes: System and Component. A system consists of components and subsystems. The specific components of the hydro-thermal power systems are all derived from the Component class, via intermediate classes that add some functionality, eg. the RiverComponent class that defines a pointer to the next component in the river (Figure 5.3).

![UML diagram illustrating the main features of the hydro-thermal power system.](image)

Figure 5.3. A UML diagram illustrating the main features of the hydro-thermal power system.

5.1.2 The Scenario Tree

A general, basic tree class was written in Java. The scenario tree implementation added support for associating a probability with each node in the tree, and methods for fast access to nodes at the same level in the tree, which helps calculating expected value, percentiles etc.
5  Implementation

Operations are not performed on nodes directly, instead using iterators to keep the containing tree instance up to date of any changes (Figure 5.4). For example, when adding a child node, there is a need to update the list of nodes at that specific level in the tree.

![Diagram of tree structure with classes and methods]

Figure 5.4. The scenario tree class inherits a more general tree class.

5.2  Results

The implemented prototype consists of about 200 Java classes, where one third are specific to the SPOT user interface, and the rest is more general and could be used in other projects. The functionality described in the design phase was implemented.

5.2.1  Problem View

The Problem view is dominated by a display of the system, and the properties of the currently selected component. There are tabs for selecting which part of the hydro-thermal power system to view (for instance, a river or the thermal plants) (Figure 5.5). A list of available problems can be extracted from the left side of the window.
5 Implementation

Figure 5.5. The Problem view with a river displayed to the left, and some of the properties of selected component “Station 2” visible to the right. Illegal values (out of bounds) are marked with red colour, legal but non-default are marked with green.

In the bottom of the window, the property that is currently selected is plotted in a time-series diagram (Figure 5.6).

Figure 5.6. Minimum generation of “Station 3” plotted as a time-series.

There are several ways of viewing the system, including a geographic view, as illustrated in Figure 5.7.
5 Implementation

Figure 5.7. Viewing the river from Figure 5.5, this time using a geographic representation.

5.2.2 Solution View

After switching to the Solution view, the scenario tree is plotted both in a miniature view displaying the whole tree, and in a larger, more detailed view. What part of the tree that is displayed in the more detailed view is indicated in the miniature view. When a time period is selected, the corresponding probability distribution is plotted to the right (Figure 5.8).
Figure 5.8. The Solution view. Week 20 is selected and the probability distribution of the nodes at the corresponding level in the tree is plotted in the histogram to the right.

By pressing a button, mean and percentiles can be plotted. Mean, along with the standard deviation, is drawn as green lines (Figure 5.9) both in the scenario tree view and the distribution view.

Figure 5.9. Mean and standard deviation plotted on top of the scenario tree.
5 Implementation

Percentiles can also be drawn on top of the scenario tree and the probability distribution plot of the currently selected level. One of them is the 50% percentile, i.e. the median, and is drawn in blue. The two others are set using a slider or a text field. The area between these percentiles are then drawn using a semitransparent yellow color. It is also possible to hide the tree and view only the percentiles along with the contour of the tree (Figure 5.10).

![Figure 5.10. Outlines of the scenario tree along with the 75%, 50% and 25% percentiles are plotted.](image)

5.2.3 Evaluation

Anna Höök and Johan Andersson, the two seasonal planners at Fortum who formed the user group, were shown the implemented prototype. It was first demonstrated informally feature by feature, and the workflow was explained. I made clear that interrupting me for questions or clarifications was appreciated. Following the demonstration, the details of the interface were discussed.

They wished for a way of specifying temporary shutdown of a power station. Currently one could perform a shutdown by setting maximum production to 0, but the users wanted an explicit way to do this. The question about shutting a part of the station down was raised, for example when one of the turbines is being repaired. However, it is currently not clear how this is handled in the model.

Some sort of overview over all stations was also wished for. It was suggested that it could be a table with some of the stations’
characteristic properties, as a complement to the current graphical schematic maps. This functionality existed in KR95, and was thought to be useful.

The users remarked that they missed some station properties, like height of fall and owner share. These were however currently not included in the model, and how they should be handled is under discussion.

The main criticism regarding the Solution view was that there was no way of seeing the connection between the spot price and inflow of the currently selected power station. An improvement would be to display the values of these variables when selecting a node in the tree.

Overall the users seemed satisfied with the prototype. However, a discussion emerged regarding the difficulty of choosing one plan when there are several scenarios. This led us into the question of how SPOT is going to be used and how it will be incorporated in the organization.
6 Discussion

Here follow some of my personal reflections on the results of the design and implementation of the user interface. I will first briefly discuss the general findings of the project, and then give some recommendations for the development of a decision support system based on a stochastic optimization model. Finally I describe the planned future work of the SPOT user interface.

6.1 General Findings

Separating the user interface in two logical views, problem and solution, allowed for a straightforward visualization of the features of the mathematical model. While abstracting the computational kernel as a black box whose interior is unknown, it still gives the user an indication of how the system works. It gave the user a natural hint on how to use it, and what to focus on: first you set up a problem, then you evaluate its solution.

The hydro-thermal power system that is part of the definition of a problem was represented by a schematic and a geographic map, as a result of “recognition is easier than recall”. It was complemented with a fast selection mechanism, where the user can type the first letters of a component’s name to select it. When evaluating the implemented prototype, the user group missed an overview over all components in the system, preferably in the form of a table with names and characteristic properties, as it was what they were used to in the current tool.

The scenario tree that represents the solution to a problem is plotted as a two-dimensional time-series that can be interactively evaluated. Based on the users reactions in the design phase, this was the preferred way of doing it. A three-dimensional view of the scenario tree would allow for visualizing the probabilities of the nodes in the same diagram, but would significantly add to the complexity of the presentation.

Regarding the implementation, the connection between the user interface and the server was not established during the project.
Therefore, the believed advantages and disadvantages of the client-server solution could not be verified. Suffice to say that care must be taken so that the connection is robust, secure and does not in a significant way hinder the use of the tool. The security aspect is likely to be the most important if the server is run on a machine outside the domains of the power producer, for example as a subscription service.

### 6.2 Recommendations

A typical deterministic optimization problem has only one optimal solution, whereas in the stochastic case there are several depending on the outcomes of the stochastic variables. The latter describes the real-world problem in a more realistic fashion than the former, but puts more responsibility on the shoulders of the user who is to interpret the solution. There is no longer “one right answer” so to speak. Therefore it is of utmost importance that the problem and its solution is visualized in a way where the stochasticity of the problem is emphasized.

It is also important to stress that the results are a representation of reality, a model. It is most likely that the actual outcome does not exactly match any of the scenarios computed. By choosing a sound metaphor for the user interface, this fact may be accentuated.

There are many ways of visualizing the problem and its solution, i.e. there are several perspectives to use when viewing the data. One perspective may be better suited when looking for a specific feature of the data, while a different perspective may be the best in another situation. Therefore, implementing only one of these perspectives is not recommended. The dialogue with the user should be central during the project, as it is difficult for the developer to know what the user will be looking for in the data. Requests for new perspectives may (and most certainly will) arise during the project, indicating that the platform of the user interface should be flexible and modular, to allow extending it with new perspectives easily. The implementation of a new perspective should not trigger a complete rethinking of the design.

Other software such as Matlab and Microsoft Excel contain many different methods to visualize data. Therefore, the ability to export data
6 Discussion

to these programs could be very valuable, or at least providing a function to export data to a more general format that can be imported in the programs.

Implementing the user interface in Java as a client and the computational kernel in C++ as a server is a solution that should be considered. This combines the fast development of graphical applications in Java with the computational speed of a C++ program into a powerful platform independent package.

6.3 The Future of SPOT

The future work on the SPOT user interface will focus on such topics as:

- How can one make the connection between different parameters such as spot price and generation at a certain power station clearer?
- How will the last week’s observations be imported into the user interface? Currently, this data typically arrives in the form of Excel files, but there were plans of implementing a database system at Fortum for this purpose.
- What control over the generation of the resulting scenario tree will the user have (for example adjusting for spot price and inflow predictions)?
- In what way will SPOT be used in seasonal planning? This question was raised during the evaluation of the prototype, and is of high importance.

Regarding the remaining financial parts of the model, for instance the contract portfolio, it leads us into a different question altogether: how will SPOT be used in an organizational perspective? At Fortum, the seasonal planners and financial analysts use different tools and techniques, and their findings are then discussed and merged in a weekly meeting. SPOT combines seasonal and financial planning, which means that one must think about how to incorporate it in the organisation in order to take full advantage of its capabilities. This would mean changing the routines and responsibilities of the different divisions involved in the seasonal planning at Fortum.
7 Conclusions

A user interface design of a decision support system for seasonal production planning in a hydro-thermal power system was developed, and a prototype was implemented in the programming language Java. Thus the goals of the project were reached.

The user interface was divided into two main views, the Problem view and the Solution view, in which the user sets up a problem and evaluates its solution respectively. A graphical representation of the hydro-thermal power system, and the properties of the currently selected component, dominates the Problem view. In the Solution view the scenario tree that represents the solution to the problem is plotted, and can be evaluated interactively.

The user group that participated in an evaluation of the design and the implemented prototype was satisfied with the result. Based on its suggestions, the continuing work on the user interface will include finding a way to visualize the connection between different parameters in the model. But most important is implementing and testing the communication between the separate user interface and computational kernel.

An imperative question that remains to be answered is: how will SPOT be used in seasonal planning? The future of SPOT may depend on how it is incorporated in an organization.
8 Acknowledgements

I am grateful to a number of people that have been helpful during my master’s project. First and foremost Stefan Feltenmark, my supervisor at Optimization Partner, deserves a thousand thanks for the support and helpful suggestions he has given to me throughout the work.

I would also like to thank Anna Höök and Johan Andersson of the user group at Fortum who took the time to participate in the interviews.

Thanks to the rest of the SPOT team: Jan Holst, Roger Halldin, and Ann-Brith Strömberg, who gave a lot of valuable input and answered my questions with patience.

Thanks also to Ulf Brännlund at Optimization Partner for all good advice and cheering me on in general.

Finally, I would like to thank my supervisor at KTH, Lars Kjelldahl, for the support.

Thank you all!
9 References


References


10 Appendix

10.1 User Interviews
User interviews were conducted in Swedish. A plan was prepared for the requirements gathering interview. Provided here is an English translation of that plan.

10.1.1 Requirements Gathering Interview Plan
1. Describe the organization in which you work!
2. Describe a typical week at work!
3. What tools do you use in your work?
4. In what form do you get input data like observations, input-output functions etc.?
5. Do you perform any portfolio management?
6. Describe KR95 and the workflow when using it!
7. What data does KR95 need?
8. What is the acceptable time limit for a solution to be calculated when using KR95?
9. Is there a need for simulating extreme scenarios, for example flooding?
10. What level of detail do you want?
11. What do you prefer: being able to compare two solutions or having a more detailed view of a single solution?
12. What is the better of a realistic solution taking hours to calculate, or a rough solution taking a few minutes?
13. What language is preferred (Swedish, English...)?
14. Other suggestions and opinions?