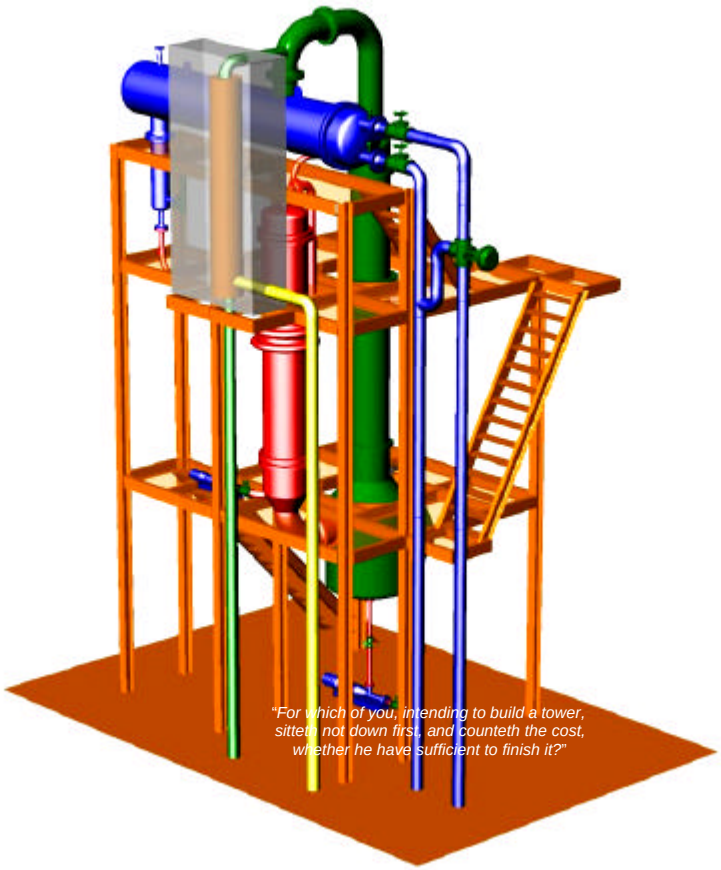
LIFE CYCLE

SIMULATION

Using

BAYESIAN



CALCULATION

Final Project Commissions:

Dr.Drs. L.J.M.Rothkrantz
Prof.Dr. H.Koppelaar
Dr. M.van Wagenberg
Dr.Ir. E.J.H.Kerckhoffs

F. Heru Utomo
January 2001

TU Delft
Delft University of

Faculty of Information Technology and Systems
Knowledge-Based Systems

TABLE OF CONTENTS

ABSTRACT

This thesis, that is produced as a last step in the program to obtain a master degree in Computer Science, at the department Knowledge-Based Systems, Technical University of Delft, covers the Life Cycle Simulation in the Process Industry. It covers two main areas: 1) apply the Bayesian approach for assessing the degradation of equipment of process plants and derive applicable knowledge from the results; and 2) develop a data model that facilitates the intake of diverse data from different sources: the process applications with which the chemical processes are designed and projected as a prior expectation and the real time process management and ERP-applications that provide the details with which prior estimates can be checked, thus providing the outcome of the Bayesian assessment. As a result, an application has been designed, that incorporates the different aspects of the thesis.

The thesis provides feedback on a chemical process that is analyzed. It shows the logic of the modeling that is needed, when applying Bayesian calculation. As well, the Bayes' approach and how it can be applied in a plant environment is discussed. It is concluded that this approach seems to be fit for assessing degradation patterns of equipment, to express that degradation in lost revenue over the years to come and to derive meaningful advice to the organization in terms of: change maintenance patterns or replace equipment items due to excessive degradation.

To apply this Bayes' approach in a practical setting, the data streams of different applications have to be imported and prepared for the simulation. In this thesis, the data model that has to support the simulation and the flexibility that is needed to be able to integrate new data, are discussed. It has been illustrated that a hierarchical object oriented set-up provides a logical data model and that from the model, data can be retrieved in a flexible way, so the Bayes' approach can be changed and fed with new variables at will.

The thesis then highlights the way the interface is used to derive helpful conclusions to the user of the simulation. The principles on how data should be provided are discussed and it is explained how this guidance is applied for this particular interface. In this part, it is as well illustrated that the usefulness of the application is enhanced by abstracting knowledge from the Bayesian calculations; and it is visualized how to provide this as a consult to the end-user.

In the final part of the thesis, the effectiveness of the Bayes approach in the specific plant setting, a process that is run at the University of Delft as a test-bed, is analyzed

and it is discussed, how this approach would fit a larger chemical process. Thus, the thesis ends with some recommendations in the area of applying the Bayes approach, in further developing the software model and on how to derive advise to the regular end-user of the application.

ACKNOWLEDGEMENT

THE BASIS FOR THE LIFE CYCLE THESIS

Several years ago, I was contacted to work on a simulation project, assessing organizational functioning, based on organizational functioning. Secondly, during the Real Time Artificial Intelligence course, I was involved in assessing Personal Moods, derived from facial expressions. The first project subsequently introduced me to the Bayes approach, the second project to Intelligent Reasoning as a part of Knowledge Systems. Both approaches kept me interested and thus let to the choice to plan the final thesis in this field of Knowledge Systems. Having worked on Seaview's Intelligent Organizational Performance Assessment, this company provided an opportunity to apply the Bayesian Approach towards assessing Life Cycle Performance.
Seaview B.V. is a small-specialized research and development company, providing management information services to the Process Industry. As a part of their future services, a research program had been set-up to develop a simulation that could be used for assessing work-processes in Engineering and Maintenance. Within this program, my thesis was positioned, this lead to defining the Life Cycle Simulation, based on the Life Cycle concept in the Process-Industry. The Life Cycle Concept is, as will be explained, a rapidly developing topic, aimed at controlling and decreasing cost and increasing benefits during the more then 20 year Life span of a process plant.

When starting the thesis, I was stationed at the API Institute at the Delft University of Technology. This institute is dedicated to the Process Industry. Being an institute, affiliated to the Department of Chemical Engineering as well as to Mechanical Engineering and performing Applied research to the industry, this department has several process installations that are designed and monitored by state of the art software applications. One of these installations, based on a distillation process, was selected. With regard to this installation, the Life Cycle Simulation should be fine-tuned and applied.

ACKNOWLEDGMENT OF SUPPORT FOR THE LIFE CYCLE THESIS

I would like to acknowledge Dr. Dr. L.J.M. Rothkrantz of the Computer Science Department of Delft University of Technology for providing all the academic support in designing the Life Cycle Simulation application and in assisting in checking on

alternative applications that might fit modeling purposes. As well, I would like to thank Dr. F. Groen, affiliated to Seaview, for assisting in understanding the initial set-up of Bayesian modeling.
I have received great support at the API Institute. Therefore, I would like to thank Prof. Dr. J. Dhillon for arranging the hospitality at the Institute and Prof. Dr. J. de Graauw and Dr. Ir. F. Fakhri for helping me to understand Chemical Process Simulation and the specific Chemical Installation I was working on. Furthermore, I would like to thank Ir. Chrismono for assisting me in understanding Aspen and the reach of this application and Ir. A. Mohammed Ali for preparing 3D models of the installation to clarify the different items. During the cause of this thesis, we have been in touch with Gist-DSM and I would like to thank Ir. A. Ligtenbarg for specific remarks on aspects that are of importance to applying the Life Cycle Simulation.
Finally, I would like to thank Dr. M. van Wagenberg of Seaview B.V. in providing this opportunity to work on this research project and in providing practical advise on how the Simulation should be conceived and used in a practical petro-chemical environment.

Delft, December 24, 2000

1 INTRODUCTION

1.1 THE SIMULATION OF LIFE CYCLE

The idea of Life Cycle is not a new invention. A very popular book from the first century has noted "*For which of you, intending to build a tower, sitteth not down first, and counteth the cost, whether he have sufficient to finish it?*" Thus the main idea of the Life Cycle is to count every financial detail before one beginning to work his *dream* out.

In this report we will limit our idea of Life Cycle to the process industrial term. The process industries bring one of more products. In a process there are several specific pieces of equipment used. These pieces of equipment perform one or more processes and form a plant. Each piece of equipment has its own life time expectancy and its own particular maintenance techniques. There come times when these pieces of equipment require maintenance jobs to keep it up to standard performance and at last there are times when these pieces of equipment cannot be used anymore and must be replaced.

1.1.1 THE IDEA

Based on experience in Engineering-Management of Process plants Seaview started to develop a simulation approach for the Life Cycle decisions with regards to maintenance and revamps.

This inclination was based on questions from O/O's[1] and ECP's[2] with regards to how to manage Life Cycle Engineering and Maintenance decisions. Since more and more plants are designed and managed to serve a more flexible life cycle in terms of:

- More pro-active maintenance: revamp if needed because of degradation parts
- More pro-active revamping because of need for new processes and efficiency
- More focus on risk-evasive execution of revamps and upgrades.

[1] O/O is the abbreviation of Owner/Operator. Companies like Shell, AKZO and Dupont are owner of plants and they run also this system.

[2] ERP is the abbreviation of Enterprise Resource Planning. These are applications like BAAN, SAP and PEOPLESOFT. These application maintain the resources of a company: logistics, administration, human resources, planning and budgeting.

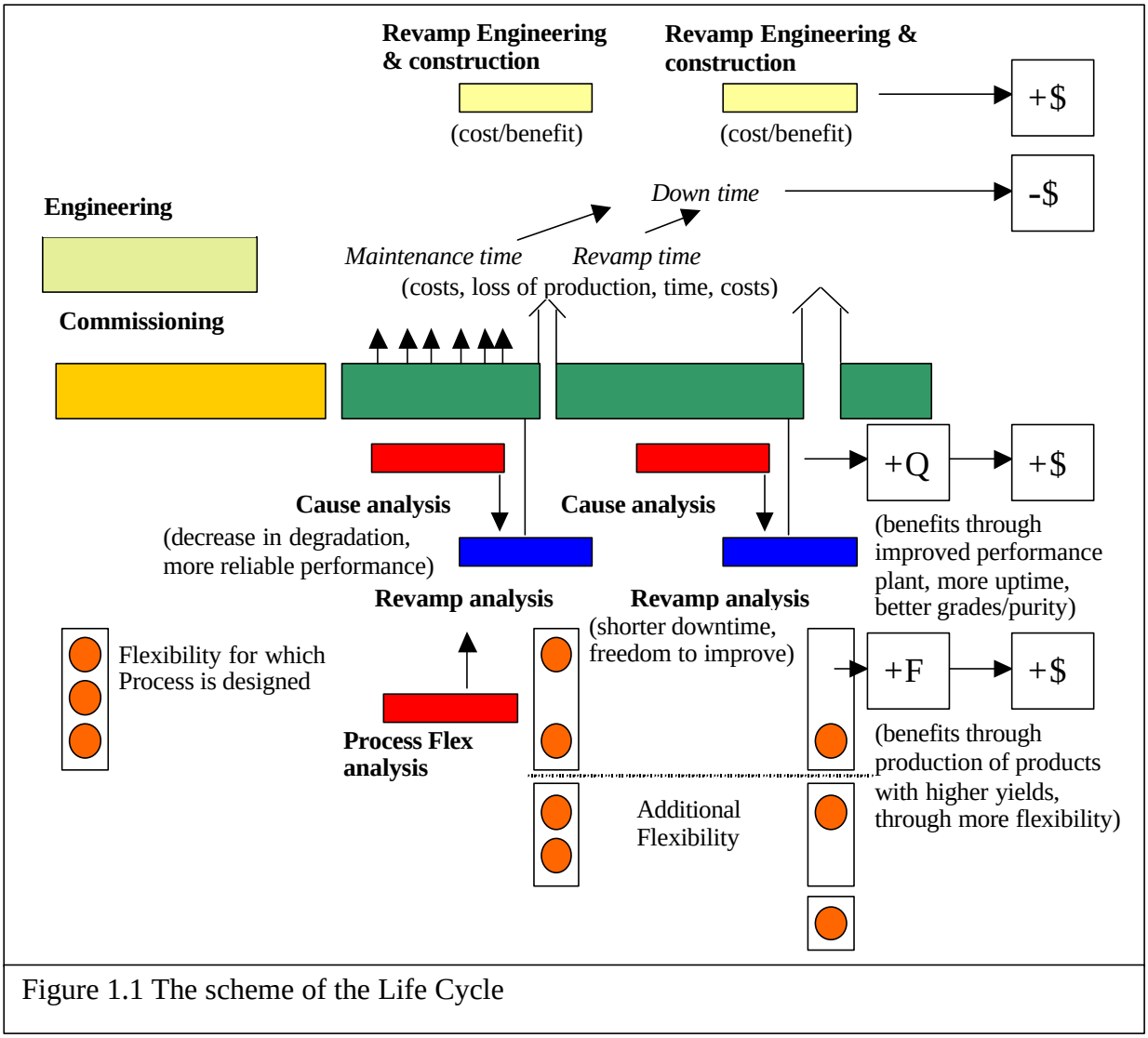


Figure 1.1 The scheme of the Life Cycle

Based on these premises, we developed a simulation approach that is based on a newly explored innovative uncertainty calculation, the Bayesian approach. Based on this approach we provide a simulation support for:

- Assessing degradation of equipment and impact on performance and yields
- Decisions on whether to improve maintenance management or decide on revamps
- Assessing details of the revamp-project, especially the organization between partners.

All these aspects can be translated into flexibility, money and time/downtime as is illustrated in the model (see figure 1.1). This model depiction signifies the costs and the benefits of the Life Cycle concept.

1. Revamps imply downtime and investment costs ("- $")
2. Revamp- and cause analysis will lead to improvements, thus benefits ("+$")
3. New process flexibility will lead to benefits ("+$")

4. This has to be offset against downtime ("-$")

What has to be factored in, is:
- Probability of extended downtime, a risk factor in terms of costs
- This risk factor needs to be addressed; the lower variance of Revamp Lead time, the higher probable benefits.

In this report, we will depict one of the phases of the simulation, the simulation of a distillation column, a reboiler, a membrane and pump. This micro-process illustrates abilities to use concurrently ASPEN[3] data, SAP data, TDC data and other data that can serve as an indication of the performance of the process. As well, it illustrates the ability to translate the assessment in effects on financial yield

1.1.2 THE PROCESS LIFE CYCLE

Every process and related installation have a life cycle that is based on a process design. Since ASPEN is commonly used in chemical process industries, for the sake of argument we assume that the prior assessment for the performance and yields are based on ASPEN.

The process is engineered, built and commissioned. The first operational results will be metered by a process control system, for example a Honeywell TDC system. Thus, a prior expectation can be checked with some results with regards to the performance of the process. Similarly, we assume SAP data that represent Process Costs, Maintenance Costs and Effectiveness. These results are:
- The actual indicators of performance as measured by the TDC system
- The actual costs of activities that were needed as measured by SAP.

[3] ASPEN is a process steady-state simulation program. It calculates the results of a process for any given parameters.

These prior data and actual data are then fed into the simulation calculation and the expectations with regards to yields or performance are corrected. These new expectations can, as with prior assessments expressed in monetary benefits. Sometimes, the degradation will show to be much more prevalent than expected, thus other maintenance procedures and revamps have to be discussed. The depiction of the impact in yield will help managers in setting priorities.

The assessment can be done on an item-by-item level. Usually, it will be a 3-phase approach that will work bottom-up:

1. On an equipment by equipment level.
2. On a sub-system/process-level.
3. On a plant level. (figure 1.2 shows a picture of a laboratory scale plant)

First, the process that is used as an example will be illustrated (see figure 1.1). Then the logical simulation models of the pieces of equipment are illustrated. Next, details will be shown.

This sub-system shows a column and condenser, a reboiler, a membrane and a pump. The reboiler may show degradation that is the responsibility of maintenance. However, at a certain level of degradation, other aspects have to be taken into account that may lead to considering a revamp. Columns are fairly stable in terms of degradation. Columns act only as container of the process. These columns do not exchange heat and therefore the column itself degrades very little. Thus, it is modeled as a part of the reboiler. The membrane has a history of degradation that impacts maintenance and requires revamps. The pump may have several problems that may or may not lead to replacement.

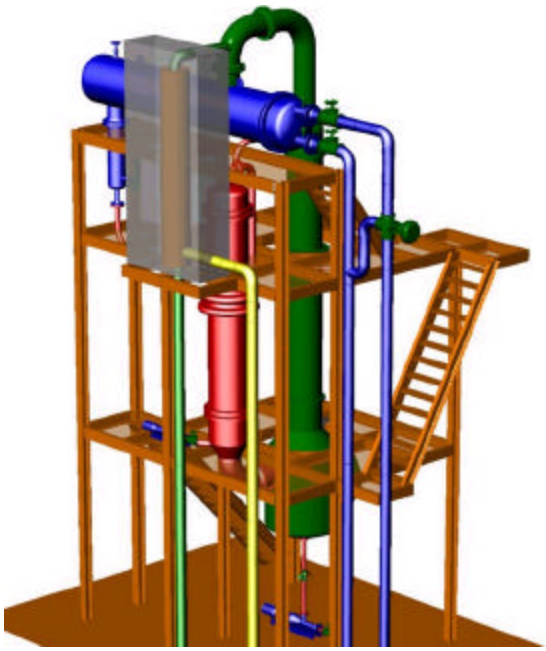


Figure 1.2 A picture of a laboratory scale plant

1.1.3 LIFE CYCLE SIMULATION: A COMBINATION OF PROCESS- AND BUSINESS SIMULATION

Since many managers are aware of improvements in the area of process simulation, we will explain the development by pointing at that development and our development of Bayesnet Simulation.

Process simulation has made advances in the last years. De-bottlenecking simulation, Real-time link-ups between Process simulation and Process-control and other forms of integration has been successfully applied. These process-based simulations address process improvements, as well costs and yields and are, therefore, of utmost value to O/O's and ECP's.

The Bayesnet Simulation as described above does not attempt to improve processes or the way the processes are managed. It deals with the question whether managers should be alerted on changes that affect Life-Cycle costs and benefits. This is based on uncertainties, not on a set of well-proven models between process design and actual process functioning and on which aspects of the process can be managed. The Bayesian approach to uncertainty is unique with respect to:
- That a lower set of certainty can be addressed
- More diverse data can be combined

And therefore considered to be a better early-warning system than other approaches[4]. When alerted by the Bayesnet Simulation that is more business oriented, the Process Simulation systems as mentioned would enhance the improvement steps that are to be taken.

Furthermore, in our Bayesnet Simulation, we address the functioning of the Maintenance and Revamp-Organization. Especially, the way the proven quality of the organization is going to impact the level of uncertainty of lead-time of major revamps, is an important step. The Bayesnet Simulation will tell management what level of risk towards stretched lead-times will be linked to the revamp and which organizational improvements may decrease this risk for prolonged lead-times. This part of the simulation is as well unique. These aspects of Life Cycle decisions, together with advanced Process oriented simulations will provide the following valuable combination: the advanced Process Simulation provides details on how to fine-tune the process and the Bayesnet Simulation provides insight in the business-risks and engineering-uncertainty. Together, they support the Life Cycle perspective and strengthen decision-making and increase versatility and efficiency. This integrated decision making seems to be an important element, when one wants to reap the benefits of the Life Cycle approach: provide an environment that will decrease as much as 30% of the total Life Cycle accumulated costs.

By providing as well technical as managerial inroads into the data, we assume that the simulation will be concurrently used by disciplines and by management. Disciplines may be more inclined to view the simulation from a technical angle as illustrated

[4] Other techniques like Fuzzy Logic does not represent the uncertainty naturally, an uncertainty factor in the real system can be changed based on facts. Another AI techniques like neural network needs a lot of training and therefore offer less flexibility for many processes.

above and management more from a cost/lead-time angle. By providing different angles to simulation and providing cross-pollination between technical simulation and business simulation, this Bayesnet Simulation enables proper Life-Cycle decision making.

1.2 DESCRIPTION OF FINAL PROJECT MASTER THESIS

This project will initiate the Life Cycle simulation using the Bayesian approach. In this project a working simulation will be built applied to a process plant. The application of this simulation will be applied onto a small process plant that was built in the Laboratory of Process Equipment at the TU Delft.

In this simulation application project includes:

- Modeling of process into a Bayesian network
- Modeling of a process database for this simulation
- Applying the simulation to the available measurement data on this process.
- Building a software prototype for this simulation

1.3 CONTENT OF DOCUMENT

This report will begin with a short description of this simulation process we want to apply. The process will be explained in chapter two. Chapter three will provide give a literature review about the Bayesian techniques. Chapter four describes the global explanation of the simulation application. Chapter five discusses the modeling process for the Bayesian model. The next chapter we will take a look at the database modeling process. Chapter seven shows us the result of the simulation tests and the discussions. Chapter eight will give the conclusion of this report.

2 AN APPLICATION OF THE SIMULATION

2.1 HYBRID DISTILLATION / VAPOR PERMEATION PROCESS

The separation process is a process that separates a mixture into two or more products. For example a mixture of water and alcohol and as products: pure water and pure alcohol. There are several methods of such separation, but the two most common ones are distillation process and crystallization.

The Laboratory of Process Equipment at the TU Delft is doing not only researches on those two kinds of processes but also the so-called *vapor permeation process*. That is

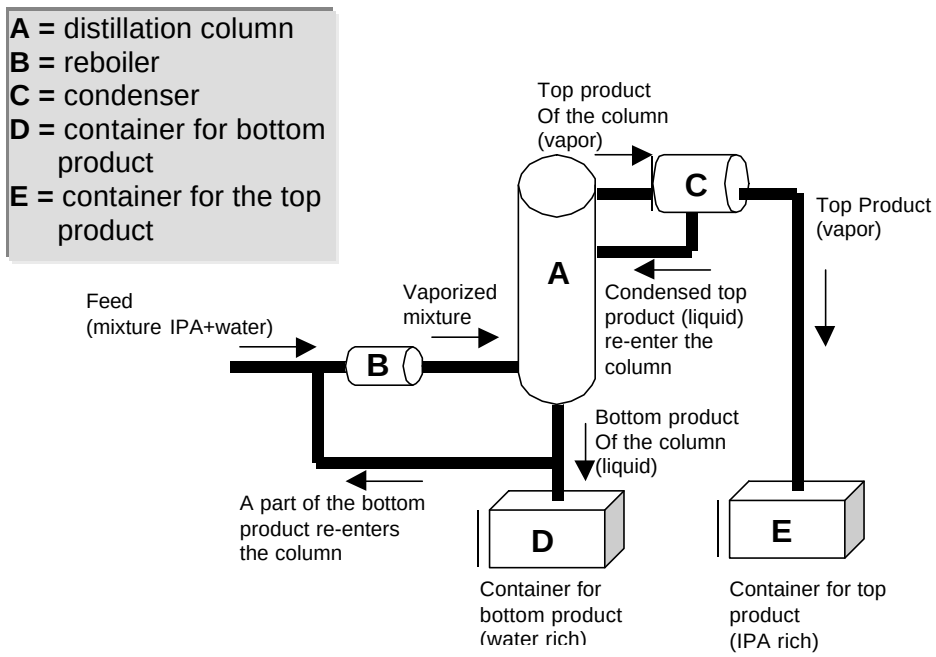


Figure 2.1 A schematic view of a distillation process

a process which feeds vapor through a membrane unit. In the next section we will take a brief look at the distillation process and the vapor permeation process.

2.1.1 DISTILLATION PROCESS

Distillation technique is the best-established technique in the process industry. This technique covers 90% of all separation processes, due to its simplicity, reliability and well-known design procedures.

The idea of this process is heating the mixture to a certain temperature (see figure 2.1). Two different kinds of liquid will vaporize at different temperatures. For example water will vaporize at 100^{o}C and alcohol will vaporize at about 80^{o}C. This means that if we set the appropriate temperature for the alcohol to vaporize beyond 80 degrees Celsius and less than 100 degrees Celsius, the condense that develops has a high alcohol content. Of course when we choose a temperature not all alcohol will instantly vaporize. Secondly, some water will vaporize, which means that one does not get pure alcohol at the top of the column and pure water at the bottom of the column in one cycle. The product, which is at the top of the column, will re-enter the column as *reflux*. This process will repeat itself in several stages until the optimum quality of product is reach. A steady-state simulation program such as ASPEN can also simulate this process. This program helps the designer to calculate the appropriate temperature, pressure and other parameters.

2.1.2 VAPOR PERMEATION PROCESS

The separation principle of vapor permeation is based on the selective mass transport through the membrane due to the difference of solution and diffusion of compounds in the membrane. This method lets the mixture, in the form of vapor, through the membranes. Depend on the molecule characteristic of the component of the mixture; the membrane will select only one component of the mixture. Figure 2.2 shows a schematic view of the vapor permeation process.

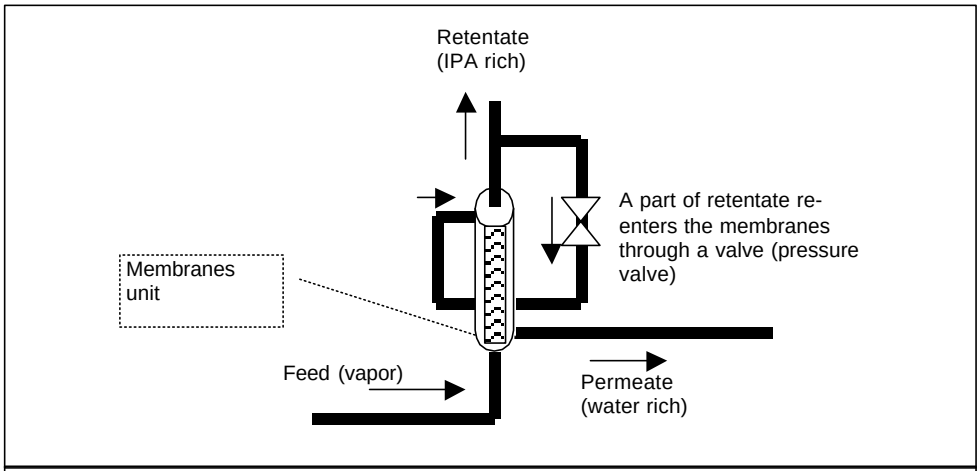


Figure 2.2 A schematic view of the vapor permeation process.

The mixture enters the membrane unit at the bottom of the unit. The pressure in the unit is very important and therefore has to be maintained by a vacuum pump. The products of this membrane unit are the *permeate* and the *retentate*. Retentate is the product that resulted from the component, which has been let through the membranes. Permeate is the product that resulted from the component that cannot (or not supposed to) pass through the membranes.

In the pilot that is built at the Laboratory for Process Equipment at the Delft University of Technology, the membranes are used to separate a vapor mixture of water and alcohol. In the retentate side it should result an alcohol rich component and at the permeate side it should result in (almost) pure water.

2.2 PILOT PLANT

The pilot plant concerning the hybrid process was researched and developed by Dr. Ir. Fahkri at the Laboratory of Process Equipment at the TU Delft in the year 1999. In this installation he has built a combination of a distillation unit and the vapor permeation unit (membranes). The purpose of this project was to break the limitation of distillation process due to the azeotropic[5] behavior of the alcohol (IPA = isopropanol). Focus of this research was to study the behavior of the membranes.

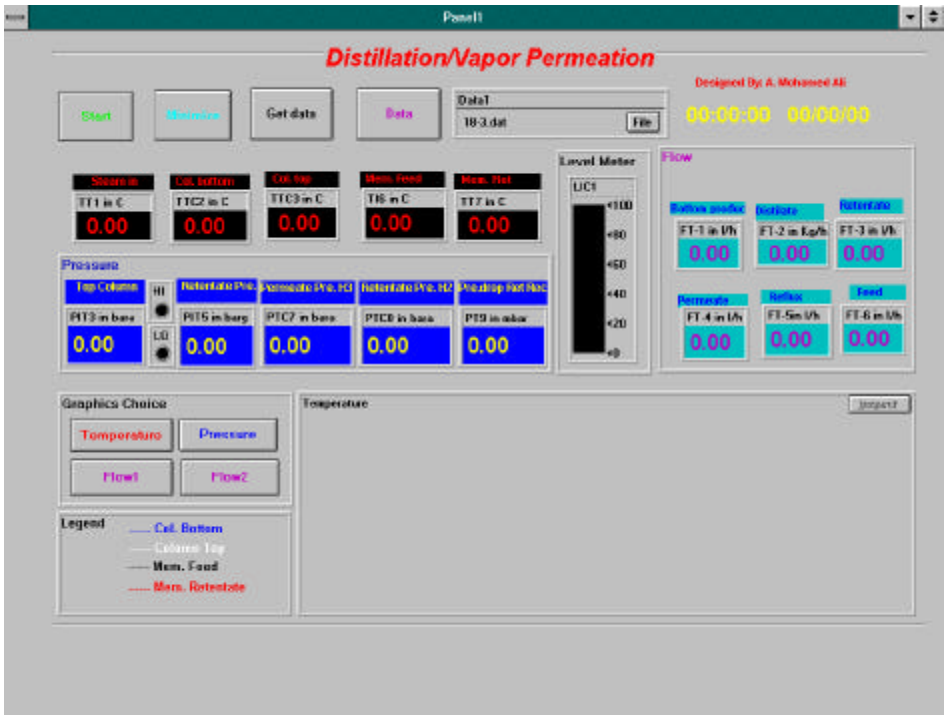


Figure 7.3 The data acquisition program TestPoint, which interface is customized for the specific process (hybrid distillation/ vapor permeation).

[5] In azeotropic processes the separation cannot reach a relatively high purity due to the characteristic of the components of the mixture.

Before this pilot plant was built it had been analytically designed with the simulation program ASPEN. This program simulates the steady state of the process, which means that it did not take the start-up condition or any disturbance into account. In other words we can say that the parameters that were calculated by that simulation are the ideal ones and thus can be used as references.

This pilot plant had several measurement instruments. These instruments were connected to a data acquisition program TestPoint on a PC. This program receives the process data online and saved them to the local disk. This program gives the freedom to the designer not only to arrange the data but also to design the user interface. This program dumped the process data in a comma-delimited ASCII file.

2.3 SETUP FOR APPLICATION OF LIFE CYCLE

In this project the Life Cycle Simulation will be applied to the pilot plant of the hybrid process. In order to begin this simulation there are steps (7) to be performed:

1. Analyze how the process works.
   First we have to study what kind of processes we are going to apply to. We have to analyze what the purpose of that process is and what are the results of the process and what are the basic principal of the processes (e.g.: physics law).

2. Collect data about the equipment, which might be important to the process.
   Our focus is on the equipments so that the next step is to study more on the behavior of the equipments, what its functions and how it works. We have to know what are the indications when a piece of equipment is degrading.

3. Create a Bayesians Network model for the process.
   After we know what the process does, how the equipment may behave and what the indicators of their degradations are, and then we can design a causal network, the Bayesian Network. To validate this design expert knowledge should be acquired and by inviting the expert to review the design. This is a crucial part of the simulation design, because we are trying to map the knowledge of an expert in term of an influence diagram.

4. Fill a database for process- and reference data from the steady state process simulation.
   To run the simulation we need a Bayesians network model, the acquired data and the reference. These pieces of information come from different software packages. They have different kinds of formats and sometimes different kinds of units. To collect the pieces of the puzzle together and match them one to another it is necessary to convert these dates into a single object database. This database includes as well the references from the acquired process data and the simulation data.

5. Create Rules.
   As an addition to this simulation the expert system rules have to be included to this simulation. These rules provide an advice based on the outcome of the Bayesians Network.

6. Implementation of the Life Cycle simulation.
   In the implementation part of the Life Cycle simulation the modules that are mentioned above will be built. The Bayesians network will be read into the simulation and all data will be used to simulate the output of the Bayesians network.

7. Testing of the Life Cycle simulation.
   At last this simulation will be reviewed and tested.

3 THE BAYESIAN TECHNIQUE

3.1 INTRODUCTION

In this chapter we will discuss the Bayes' method in the field of knowledge based systems. Expert systems itself are a part of Artificial Intelligence. Before we talk about the Bayes' method we will discuss about the background of studies in artificial intelligence (or AI).

3.1.1 ARTIFICIAL INTELLIGENCE

There is no exact definition of AI. AI itself is formally initiated in 1956. However the philosophical question about how the human learns, how the human sees, how the human thinks, remembers or reason has a 2000-year history. Most of the definitions AI have concerned with *thinking-processes* and *reasoning*. The following definitions describe AI:

> "The exciting new effort to make computers think … machines with minds, in the full and literal sense" (Haugeland, 1985)
>
> "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning.." (Bellman, 1978)
>
> "The art of creating machines that perform functions that require intelligence when performed by people" (Kurzweill, 1990)
>
> "The study of how to make computers do things at which, at the moment, people are better" (Rich and Knight, 1991)
>
> "The study of mental faculties through the use of computational models" (Charniak and McDermott, 1985)
>
> "The study of the computations that make it possible to perceive, raison, and act" (Winston, 1992)

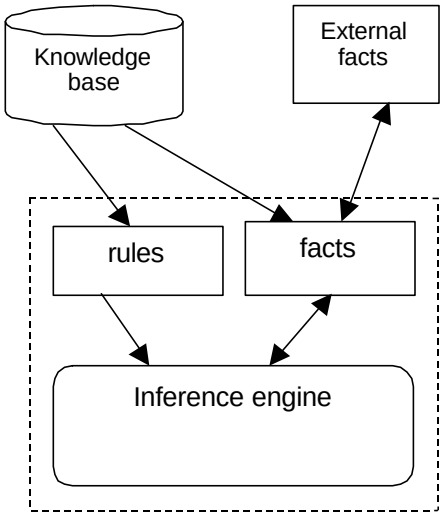


Figure 3.1 The common interface model of an expert system

> "A field of study that seeks to explain and emulate intelligent behavior in terms of computational processes" (Schalkoff, 1990)
>
> "The branch of computer science that is concerned with the automation of intelligent behavior" (Luger and Stubblefield, 1993)

Derived from these definitions, the AI research field can be categorized as focused on Automated Reasoning abilities (Bellman, Winston, Luger and Stubblefield) and on Automated Learning abilities (Bellman) and is a consequence in *reasoning system* and *learning system.* The first research group is focusing on mimicking how the human reasons. The second one is focusing on mimicking how human builds the ability to reason.

These two directions of AI correspond completely with the, *the symbolic AI* and *the sub-symbolic AI*. Symbolic AI concentrates on logic-based approaches while sub-symbolic AI concentrates on imitating the working of human's brain with it's learning capabilities.

3.1.2 EXPERT SYSTEMS

An *expert system* is a system that can act as an expert. An expert is a person who is highly skilful or has a high degree of knowledge on certain subjects. A system is a group of interacting or interdependent elements or is regarded as forming a collective entity. This definition does not imply that a computer is a human being but it indicates only the nature of the system. Thus the system is built for specific subject of expertise in some degree, meaning that a human being is still not replaceable with machines.

An expert system formalizes expert knowledge. It translates the expert knowledge into symbols (it is clear that this expert system is categorized as symbolic AI). This symbolized knowledge is then saved in a database called *knowledge base*. This knowledge base is then translated into rules. External facts are entered into the system

as information of what is now happening in the real world. The system in turn will use these informations to show its expertise through the inference engine. Figure 3.1 shows how expert system works according to [J. Jarmulak, 1999].

3.1.3 UNCERTAINTY IN EXPERT SYSTEM

Human being will always want to create something that is humanlike. Thinking rationally and logically has brought the Boolean computation, which will decide whether something is rational or not rational. This thought has become large and widely used in the modern computer until now, the binary numbers. In the binary world there is only right or wrong, true or false, ones or zeros. But then the human being has thought that to be uncertain is human.

Then the researchers have argued to conceptualize *the machine that is to act with uncertainty*, thus it will become more human. Just *true* or *false*, *ones* or *zero* is not humanlike. Just as a human, a machine cannot always have access to the whole truth about their environment. The field of AI has also expanded by bringing uncertainty into it.

There are a few techniques to bring uncertainties into expert system:

- By giving uncertainty factors (CF).
  This technique was introduced in the MYCIN expert system. In this technique each fact is given an uncertainty factor between –1 and 1. The most negative value (-1) means *false* and the most positive value means *true*. There are several methods in applying this CF into the basic logic rules (if-then-else-rules), such as techniques according to Mamdani or Gödel.

- Fuzzy logic.
  This technique was first introduced by Zadeh in 1965 [Zadeh, 1965]. Fuzzy logic seems to overlap with the first technique. In conventional *sets-theory* each element is either a member or a non-member of a set, fuzzy logic treats each element differently. An element is a member of a set in a specific degree. This value is termed the *membership-degree*, the value of this membership-degree is [0,1]. A membership-degree of an element is determined by the *membership function*, usually noted as m function.

- Statistical Probabilities.
  Probability is one of the best ways to express uncertainty. The probability theory is considered to be a part of Statistics or Statistic Computation. Statistic computation itself is based on facts, symbolized knowledge. The power of this technique is its ability to compute the *conditional* or *posterior* probabilities using the prior knowledge. In computing these posterior probabilities we are using the Bayes' formula that will be discussed in the next section. Bayes' rule allows unknown probabilities to be computed from known, stable ones.

3.1.4 JUDGING THE METHODS

The question is now which technique to be use in this project. There are several arguments support the AI engineer to choose Bayesian techniques theory over the fuzzy logic concerning the *uncertain reasoning,* according to [Rusell-Norvig, 1995]:

- Fuzzy logic misunderstood the term *uncertain reasoning*. When we want to make a proposition whether John is tall, given that he is 175cm high. The most people (American and West-European) will say 'sort of' rather than 'yes' or 'no' (*or* 'true' or 'false). This is not the uncertainty of John's height, it is not the uncertainty about saying, *"he is tall"*, since we know exactly his height already. The uncertainty that is brought here is about the term *tall* itself.

- Fuzzy logic appear because the fuzziness of the boundaries of the sets not the uncertainty of the membership. Consider there are 20 Toyotas and 80 BMW in a garage. A pick of a random car from the garage would have probability 20% that it is a Toyota and 80% that it is a BMW. In this case we are dealing with *uncertainty*. Uncertainty can be resolved by looking at the car and determining that it *is* a Toyota or it *is* a BMW, or by looking at its identities (Toyota has a logo like three joining ovals). In fuzzy logic there is no uncertainty, no further examination can make the fuzziness (or *uncertainty*, as the fuzzy logician says) be more *unfuzzy* (more certain).

- Fuzzy logic has *sinned* against the first order logic (propositional logic). In evaluating a *complex sentence* (e.g. *Green(Car) Ù Fast(Car)*) fuzzy logic use the truth value of the components:
  $T(A \wedge B) = \min(T(A), T(B))$
  $T(A \vee B) = \max(T(A), T(B))$
  $T(\neg A) = 1 - T(A)$
  In the normal propositional logic *AÙØA is* a tautology, it will always be the truth. In fuzzy logic *T(AÙØA)* is not always 1.

In contrast to the Fuzzy logic, [Russel-Norvig, 1995] summarized:

- Bayesian network represents the conditional independence information naturally.
- Bayesian network represents the joint probability distribution completely even in smaller size exponentially.
- Inference in belief networks means computing the probability distribution of a set of query variables, given a set of evidence variables. It means that in a complete Bayesian network, the network can retrieve some evidences about the environment that will influence the uncertainty in the model. The output of the model can directly be retrieved by looking at the probabilities in other nodes.
- It is possible to use approximation techniques, including stochastic simulation, to get an estimate of the true probabilities with less computation.

Due to these theoretical concepts we have chosen the Bayesian network as our technique for the Life Cycle simulation. The next section will discuss the Bayesian network technique more deeply.

3.2 THE BAYESIAN NETWORK

3.2.1 A BRIEF HISTORY OF BAYES

The Bayesian Theory was named after an Englishman Thomas Bayes (ad. 1702 – 1761). From 1731 onwards, he was a Presbyterian minister in Tunbridge Wells and was a son of the Rev. Joshua Bayes, a Nonconformist minister.

The following is quoted from the Encyclopedia Britanica:

> **Bayes, Thomas** (b. 1702, London – d.1761, Tunbridge Wells, Kent), mathematician who first used probability inductively and established a mathematical basis for probability inference (a means of calculating, from the number of times an event has not occurred, the probability that it will occur in the future trials).
>
> He set down his findings on probability in "Essay Towards Solving a Problem in the Doctrine of Chances" (1763), published posthumously in the *Philosophical Transactions of the Royal Society of London.*
>
> The only works he is known to have published in his lifetime are *Divine Benevolence,* or an *Attempt to Prove That the Principal End of the Divine Providence and Government is the Happiness of His Creatures* (1731) and *An Introduction to the Doctrine of the Analyst* (1736) which countered attacks by Bishop Berkeley on the logical foundations of Newton's calculus.

3.2.2 BASICS OF BAYES STATISTICS

Bayes' contributions are immortalized by naming a fundamental proposition in probability, called Bayes Rule, after him. The Bayes' rule is expressed:

$$P(E \mid H) = \frac{P(H \mid E).P(H)}{P(E)}$$

This formula enables us to use the prior knowledge of an event to calculate the probability of the other event. When the conditional probability of a hypothesis (H) is known, it is possible to find the conditional probability of a evidence (E). This will be illustrated later on.

This reasoning scheme contributes also in the world of science to confirm a theory [Kroes, 1996]. This rule is used to confirm a theory by means the evidences that are found that makes us belief that the theory is closer to the truth. Expressed in formula:

$P*(T \mid E*) = \frac{P(T, E*)}{P(E*)}$ and by finding a new evidence E** the new probability for T is:

$$P**(T \mid E**) = \frac{P(T, E**)}{P(E**)}$$

This calculation goes on and on reaching the true distribution of T. The disadvantage of this method is that one has to have a prior knowledge about the variable.

3.2.3 THE NETWORK

A Bayesian network is a directed acyclic graph (DAG), which nodes are events that have probabilities of occurrence and these nodes are connected with directed arrows that indicate the influence direction. Acyclic means that there is no loop allowed in the graph.

This graph can be intuitively built, one can reason when an event happens as a result of another event and then put an arrow from one node to the other. Defining the probabilities may be difficult; it required a prior knowledge of an expert or from statistical data.

The Bayesian network is unique, not like other system that has predefined inputs and outputs. Inputs of Bayesian network are **evidences**. Evidence is knowledge of an occurrence of an event. Every node of the network can be used as input and every node of the network can be chosen to be the output. Entering evidence may influence a part or the whole network values.

The Bayesian Inference Calculation

To demonstrate how the Bayesian network calculates the probabilities let us take a small case from our field of application that can be modeled into Bayesian network. Consider a distillation process and focus on the top of the column. The degradation of the condenser will be predicted here. The condenser's task is to make vapor produced in the column to condense. Because of the change in the temperature (becoming lower) the vapor will be condensed and changes into liquid. When the condenser does not work properly then it will not change the temperature as low as it should be. When this happens the flow meter measuring the distillate flow rate will show changes in the flow rate, it will become lower too. On the other hand, changes in distillate flow rate are not only caused by the condenser. When the temperature in the column is lower then less vapor will be produced. Less vapor means less distillate. The lower temperature will cause a higher bottom flow rate since more liquid is produced.

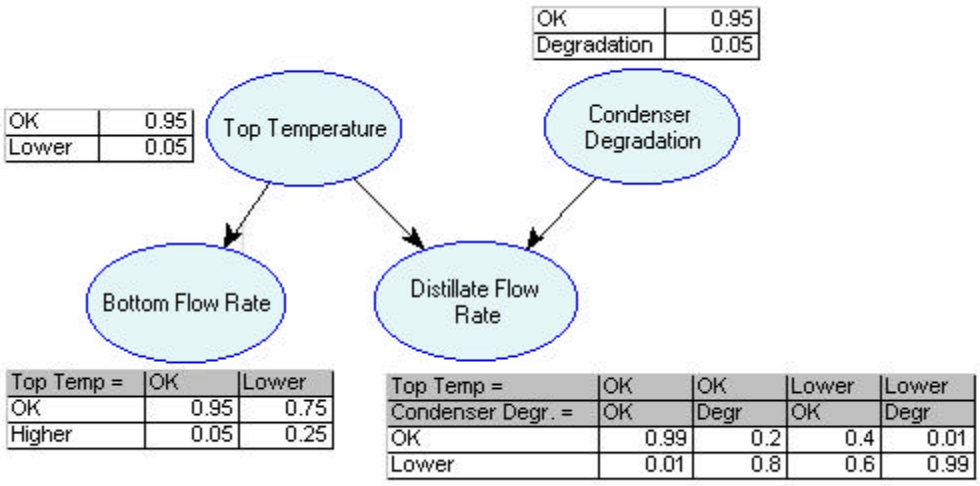


Figure 3.2 A Bayesian network model with its node definitions

The model of this simple case is illustrated in figure 3.2. The node **Top Temperature** has the states **OK** and **Lower.** We assume that this temperature will have the probability 0.95 that it is **OK** and 0.05 that it is **Lower**. The node **Condenser Degradation** has the states **OK** and **Degradation**, under the assumption that the chance that degradation will occur is 0.05. The node **Bottom Flow Rate** has the states **OK** and **Higher**, the top temperature defines the states of this node, if the top temperature is **OK** then the chance that the bottom flow rate **OK** is 0.95, but when the top temperature is **Lower** then the bottom flow rate has 0.75 the chance that it is **Higher**. The rest of the probabilities can be read at the table in figure 3.4. The tables in the figure 3.2 are the definitions for each node. When the probabilities are conditioned then the tables show the conditional probabilities and *not* the joint probabilities (joint probabilities will count up to 1).

The next step is to calculate the probabilities for each node; in order to do this the joint probabilities for each node has to be calculated. First, the **Bottom Flow Rate** node is defined as follow:

Table 3.3 The definition for node **Bottom Flow Rate**

Temperature =	OK	Lower
OK	0.95	0.25
Higher	0.05	0.75

The formula P(A | B) = P(A.B)/P(B) can also be rewritten as P(A.B) = P(A | B).P(B). In other words this formula can be used to find the joint probabilities given the conditional probabilities.

P(Temperature = OK, BottomFlowRate = OK) =
P(Temperature = OK | BottomFlowRate=OK).P(Temperature = OK)

Table 3.4 The joint probabilities for node **Bottom Flow Rate**

Temperature =	OK	Lower
OK	0.95x0.95=0.90	0.25x0.05=0.01
Higher	0.05x0.95=0.05	0.75x0.05=0.04

Summing the joint probabilities these probabilities can be calculated:

P(BottomFlowRate=OK) = 0.90 + 0.01 = 0.91
P(BottomFlowRate=Higher)=0.05+0.04 = 0.09

The joint probabilities of the node **Distillate Flow Rate** can be calculated in the same way as the node **Bottom Flow Rate**.

Table 3.5 The joint probabilities for node **Distillation Flow Rate**

Temperature =	OK	OK	Lower	Lower	
Degradation =	OK	Degradation	OK	Degradation	SUM
OK	0.89	0.01	0.02	0.00	0.92
Lower	0.01	0.04	0.03	0.002	0.08

Using a Bayesian network is about changing the probabilities (the uncertainties) given some evidences. In this case let us assume that the measurement shows that the **Distillation Flow Rate** has become lower. The simplified update rule [Perarl, 1988] for P(A | B) = P(A.B)/P(B) which evidence is found for *P(A)* is (P* is the update of P, and P** is the update of P*, and so on):

P(A.B) = P(A.B). P*(A) / P(A)*

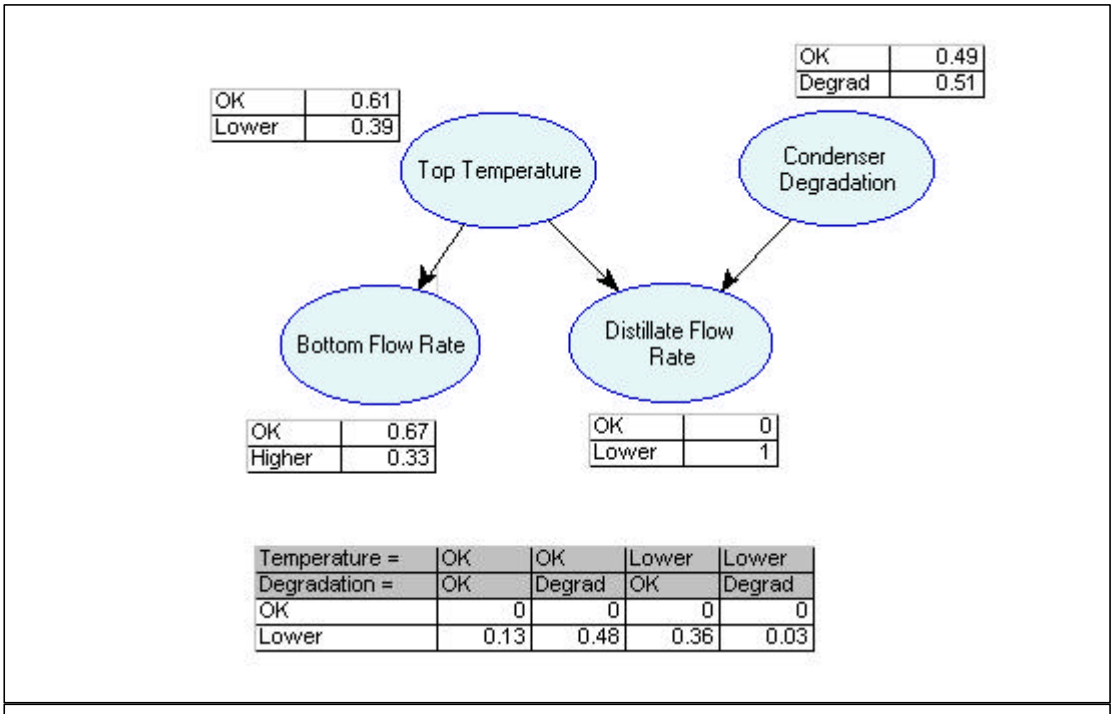


Figure 3.6 The update of the probabilities after entering the evidence.

In this example *P(DistillationFlowRate=Lower)=1* then the joint probabilities of **Distillation Flow Rate, Top Temperature,** and **Condenser Degradation** will be first updated. To update this table each probability should be multiplied with *P(DistillationFlowRate)/P*(DistillationFlowRate)*. In this case each probability will be multiplied with 0/0.92=0 for every probability from which the Distillation Flow Rate is OK and for Distillation Flow Rate is Lower each probability will be multiplied with 1/0.08 = 12.5.

Table 3.7 The update of joint probabilities of Distillation Flow Rate, Top Temperature and Condenser Degradation, given that DistillationFlowRate=Lower

Temperature =	OK	OK	Lower	Lower	
Degradation =	OK	Degradation	OK	Degradation	SUM
OK	0	0	0	0	0.00
Lower	0.13	0.48	0.36	0.03	1.00

From the table above we can easily read that the probabilities for the node **Top Temperature** is

P(TopTemperature=OK) = 0.13 + 0.48 = 0.61*
P(TopTemperature=Lower) = 0.36 + 0.03 = 0.39*

And for the node **Condenser Degradation**:

P(CondenserDegradation=OK) = 0.13 + 0.36 = 0.49*
P(CondenserDegradation=Degradation) = 0.48 + 0.03 = 0.51*

Now it is obvious that given that the distillation flow rate has the value **Lower**; the probability that the condenser has degraded has risen from 0.05 to 0.51.

Assume that the next evidence is that the **Top Temperature** has become lower. Intuitively we can reason that the condenser might not be degraded; but the temperature has caused the lower distillation flow rate. Let us see how the Bayesian network in this case will react.

Given that *P**(TopTemperature=Lower)=1* we update the joint probabilities table of Distillation Flow Rate, Top Temperature and Condenser Degradation. Each probabilities for TopTemperature=OK has to be multiplied with 1/0.39=2.56 and for TopTemperature=Lower with 0/0.61=0.

Table 3.8 The update of joint probabilities of Distillation Flow Rate, Top Temperature and Condenser Degradation, given that DistillationFlowRate=Lower

Temperature =	OK	OK	Lower	Lower	
Degradation =	OK	Degradation	OK	Degradation	SUM
OK	0	0	0	0	0.00
Lower	0	0	0.92	0.08	1.00

The update of the probabilities of the node **Condenser Degradation** can easily be read from table 3.8:

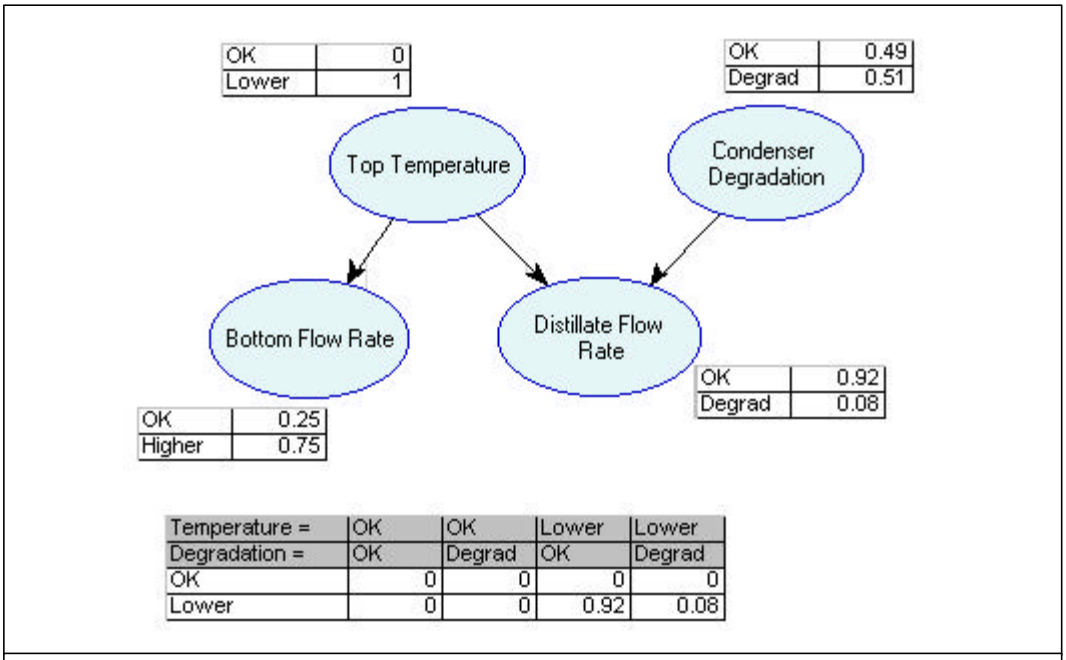


Figure 3.9 The last update given the evidence that the temperature is lower

*P**(CondenserDegradation=OK)=0.92*
*P**(CondenserDegradation=Degradation)=0.08*

This outcome has matched our intuition, which *'guesses'* that the lower distillation flow rate was not caused by the degradation of the condenser but by the lowering of the temperature.

The Bayesian calculation will then follow these rules in the simulation. Each pair of measurement will give evidences for the Bayesian network and the outcome of the network will be further used for the simulation.

3.3 TOOLS FOR THE BAYESIAN NETWORK

To apply the Bayesian Network to our Life Cycle simulation we have used two tools developed by **Decision System Laboratory (DSL)**. Decision Systems Laboratory is a research group within the Department of Information Science and Telecommunications, Intelligent Systems Program, and the Medical Informatics Training Program at the University of Pittsburgh. The mission is maintaining a research and teaching environment for researchers and students interested in the development of techniques and systems that support decision-making under uncertainty. Their methods include theoretical work, system building, and empirical studies. They rely on probabilistic, decision-theoretic, and econometric techniques combined with artificial intelligence approaches.

DSL has developed a tool named **GeNIe** and **SMILE**. GeNIe is a development environment for building graphical decision-theoretic models running under Windows operating systems. SMILE is its portable inference engine, consisting of a library of C++ classes, currently compiled for Windows, Solaris and Linux. Both programs have been developed at the Decision Systems Laboratory, University of Pittsburgh. We are making them available to the community for non-commercial research and teaching use in order to promote decision-theoretic methods in decision support systems. We have tested GeNIe and SMILE extensively and are using them in both our teaching and our research projects. We are continuously improving them and are interested in user comments. We encourage the users of GeNIe and SMILE to let us know about encountered problems and possible suggestions.

Informations about these tools can be found in their web site on the Internet *http://www2.sis.pitt.edu/~genie/*. SMILE was developed in C++ programming language, which cannot be used in the Delphi environment. In order to use these libraries we have used the so-called **SmileX**. It was developed by Ir. C.P.R. Thijssen as his master thesis at the faculty of Information Technology at Delft University of Technology [Thijssen, 1999].

SmileX is an ActiveX component that can be loaded to any development tools that support ActiveX component, this means that it can also be used in Delphi, Java, VisualBasic, Visual C++, etc. This tool is very appropriate to be used in our application since the application was developed in Delphi.

3.3.1 USING GENIE

GeNIe is a graphic-user-interfaced application that allow us visually to construct in a normal way an influence diagram i.e. the Bayesian network. Right below the menu there are the tools buttons. By clicking one of these tools we can easily draw a node and connect them to another node. A screenshot of this application is illustrated in figure 3.10.

To enter the definition of a node, just double click the node and there will appear a window that allow us to change the node name, description and probabilities.

When a network is completed one must update its values first to see the correct values. One can also right click the node to enter evidences to the node. This program saves the network configuration in its own format (*.dsl).

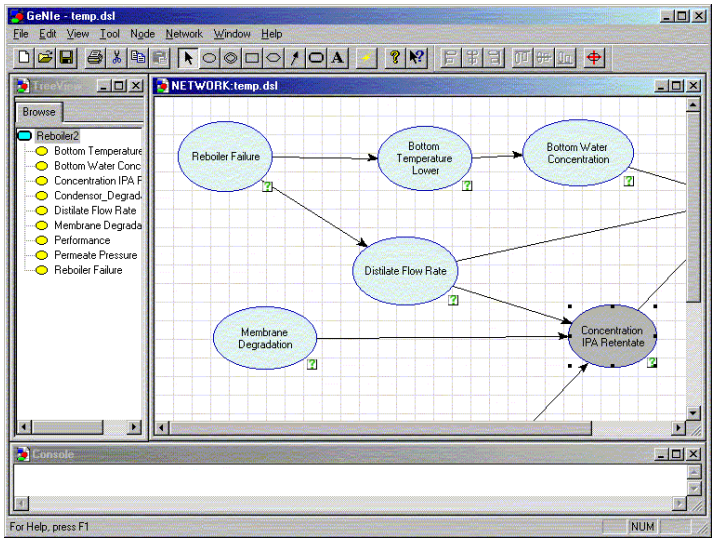


Figure 3.10 A screenshot of the simulation program GeNIe

3.3.2 USING SMILEX

SmileX is an ActiveX control that reads the network file (*.dsl) and allow the user to make some operations like entering evidences, updating the value and retrieve the current value of the nodes.

SmileX is very limited. It does not support all functions that one can have in GeNIe. It has only a few procedures and functions that updates and retrieve the value of the node. It reads the file but it cannot modify the node's definition nor save them to a new file.

Due to this limitation designing the network, it will not be supported in our simulation application, it will thus remain in GeNIe.

SmileX does not have a function that can retrieve the structure of the network, therefore the structure of the network cannot be displayed in the simulation application. It is only possible to display net nodes but without the structure. Nevertheless the structure remains, there is no need to worry that we lose the structure, it just cannot be retrieve via SmileX. The complete user manual to use SmileX can be found in the appendix.

With a few limitation of SmileX, however this tool is adequate to simulate the Bayesian network within a Delphi programming environment. It is why we have chosen this tool to be used for developing the application.

4 DESCRIPTION OF THE SIMULATION APPLICATION

4.1 THE FUNCTIONALITY OF THE APPLICATION

The Life Cycle simulation is implemented in this application. What this application simply does is taking the process data from the data acquisition computer. Before it makes a calculation for the simulation it took a reference data from a steady state simulation like ASPEN. This reference is considered as the ideal "prior" values and the calculation of the simulation is based on comparison of these values with actual process performance.

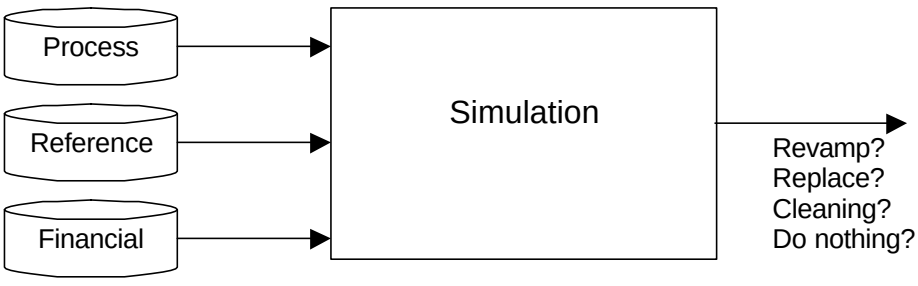


Figure 4.1 Functionality scheme of the application

The Life Cycle simulation takes not only the technical data into the simulation but as well financial- and logistical data from other sources (e.g.: SAP). The idea of this simulation is that at the end of the simulation it can provide an advice about taking a maintenance job for each equipment, replace some of the equipment or do a revamp when it calculates that revamping has more advantages than maintaining the plant.

To simulate this Life Cycle process the application is using the described Bayesian technique combined with the described rule-based technique.

4.2 INTERFACE MODEL OF THE APPLICATION

This application is built to perform an off-line simulation. The data from the process come from a data acquisition computer which records the process data on-line while the process was running. Nevertheless, in the future this simulation application will be further developed as a possible on-line simulation.

We can divide this application into three major parts: the data handling module, the Bayesians model module and the rule-based advising module. Beside these three modules there are still a few modules outside the system these modules are used to support the simulation itself.

As we can see in the figure 4.2 the process has three kinds of data that is translated and integrated into one database, the Simula database. The data-handling module handles this database. Data from this module are then used in the Bayesian model

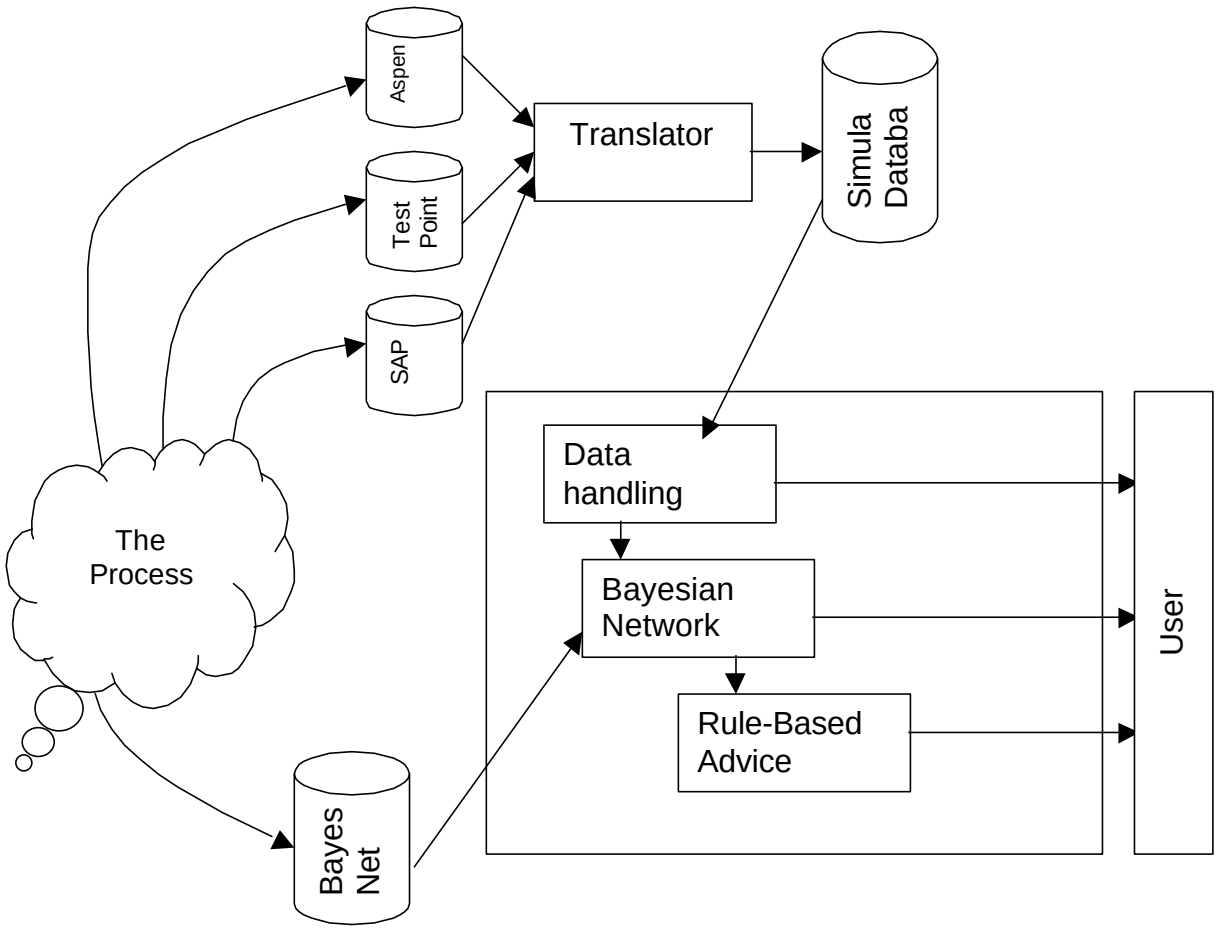


Figure 4.2 The interface of the application and its modules

module. The model of the Bayesian network is saved in a different database file to make this simulation easier to customize (the network model is built using the tool GeNIe). This module calculates the inputs and hands this output to the rule-based consulting module. This module uses the output of the Bayesian model and shows the result to the user. The user can inspect all results of the three modules.

4.3 PARTS OF THE APPLICATION

4.3.1 DATA-HANDLING

The data-handling module is used to handle the data. Data from different kinds of sources are integrated in one database. This database is not a standard database but an object-oriented one. This database is built only for this simulation and all definition of this database is included in the data-handling module.

For some reason that would be discussed in the following chapter, we would like to integrate data from the three sources, the three kinds of database those are being integrated into one database, the Simula database. These three kinds of database can be categorized as:

1. Simulation Data.
   This simulation data are produced by a simulation application that calculates the steady-state simulation program for chemical processes, called ASPEN. This program calculates the process conditions such as temperature and pressure. It simulates as well the input and the output of the process like the concentration, the mass of the mixture and its components. This program can save the data into a file that contains all information. The information is used as a reference in our model.

2. Process Data.
   Every process is usually connected to a data acquisition computer, for several purposes. These data can be used as a feedback into the controller but in this case the process data is used as experimental purposes since the installation is also built for research purposes. The form of this kind of data is usually just a table with a limited set of fields. No structure is needed for process data. The computer measured the data at a given interval, for example, every minute, every hour or each day. The computer saves the file at a given period. This data file can become very large.

3. Maintenance Data.
   The former two kinds of data have been dealing with technical data. The third kind of data is about economical factor of the process. These data come from maintenance software such as SAP, an ERP application with maintenance functionality. It contains as well data on energy-costs, usage of feedstock and the like. In this project unfortunately we cannot get data from SAP. Therefore, we will be using a dummy data set with maintenance information.

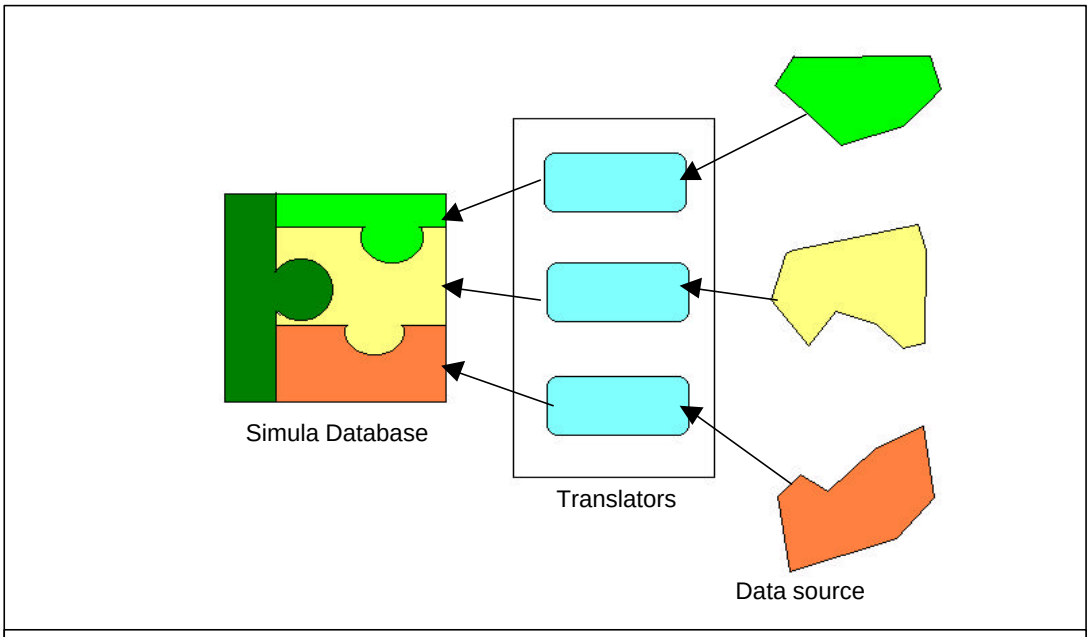


Figure 4.3 The functionality scheme of the translators

The integration of the data can be viewed as collecting pieces of puzzle. But first the pieces of puzzle have different kinds of shape that will not match to each other. Using a translator we are reshaping the pieces and then putting all pieces of the puzzle together into the structured (Simula-) database. Reshaping the pieces of the puzzle would mean recognizing elements of the data and match them with the elements of other database if they are supposed to be the same kind of data (see figure 4.3).

For example we would like to compare the data from ASPEN and from the process data that is taken from the data acquisition program. ASPEN says that temperature of the column with identity **DIST1** at the out stream **D1** is 111°C and this information should be matched with the 2nd field of the measurement data. The Simula database will read this information as **Distillate_Temperature** and this information would be matched with **miDistillate_Temperature** (mi stands for measurement index of distillate temperature). The function of this translation module is then to take the **111°C** from the **DIST1** at **D1** and fill the 100°C into variable **Distillate_Temperature** in the Simula database. Then it must fill the variable **miDistillate_Temperature** with the value 2 since this **Distillate_Temperature** corresponds with the 2nd field of the measurement data. Figure 4.4 shows how the ASPEN data looks like and figure 4.5 shows the simple structure of the process data.

This database also contains financial data from other software packages. But in this pilot project the simulation is not (yet) supported with data from software like SAP, and therefore we will momentarily use a dummy. Since this dummy is still hard coded on the program then it will have no translator module, but this information will still be provided in the Simula database.

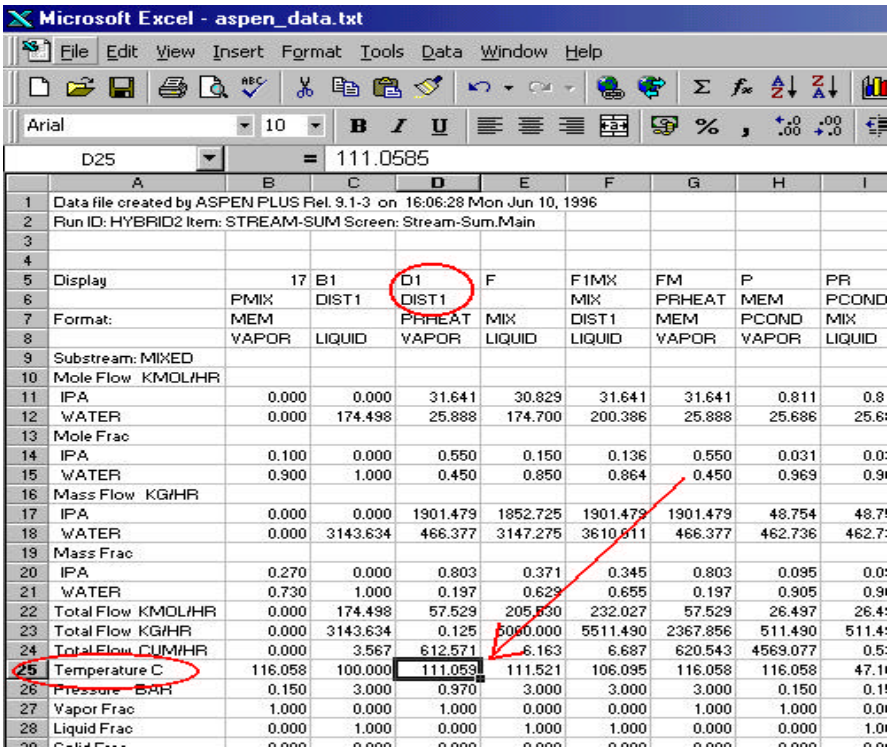
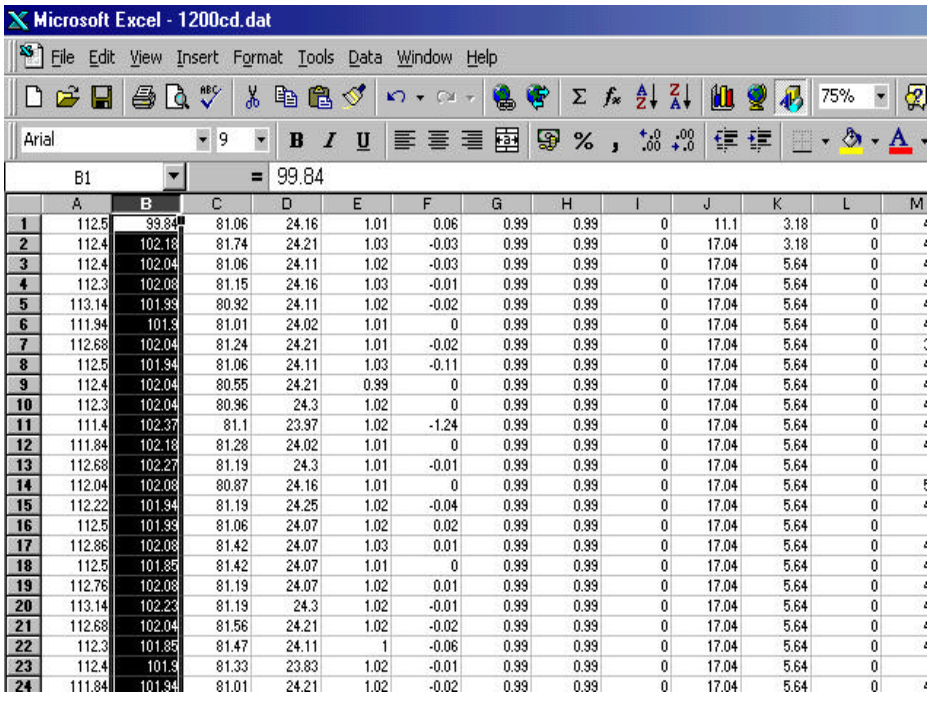


Figure 4.4 Example of an ASPEN data file.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	112.5	99.84	81.06	24.16	1.01	0.06	0.99	0.99	0	11.1	3.18	0	
2	112.4	102.18	81.74	24.21	1.03	-0.03	0.99	0.99	0	17.04	3.18	0	
3	112.4	102.04	81.06	24.11	1.02	-0.03	0.99	0.99	0	17.04	5.64	0	
4	112.3	102.08	81.15	24.16	1.03	-0.01	0.99	0.99	0	17.04	5.64	0	
5	113.14	101.99	80.92	24.11	1.02	-0.02	0.99	0.99	0	17.04	5.64	0	
6	111.94	101.9	81.01	24.02	1.01	0	0.99	0.99	0	17.04	5.64	0	
7	112.68	102.04	81.24	24.21	1.01	-0.02	0.99	0.99	0	17.04	5.64	0	
8	112.5	101.94	81.06	24.11	1.03	-0.11	0.99	0.99	0	17.04	5.64	0	
9	112.4	102.04	80.55	24.21	0.99	0	0.99	0.99	0	17.04	5.64	0	
10	112.3	102.04	80.96	24.3	1.02	0	0.99	0.99	0	17.04	5.64	0	
11	111.4	102.37	81.1	23.97	1.02	-1.24	0.99	0.99	0	17.04	5.64	0	
12	111.84	102.18	81.28	24.02	1.01	0	0.99	0.99	0	17.04	5.64	0	
13	112.68	102.27	81.19	24.3	1.01	-0.01	0.99	0.99	0	17.04	5.64	0	
14	112.04	102.08	80.87	24.16	1.01	0	0.99	0.99	0	17.04	5.64	0	
15	112.22	101.94	81.19	24.25	1.02	-0.04	0.99	0.99	0	17.04	5.64	0	
16	112.5	101.99	81.06	24.07	1.02	0.02	0.99	0.99	0	17.04	5.64	0	
17	112.86	102.08	81.42	24.07	1.03	0.01	0.99	0.99	0	17.04	5.64	0	
18	112.5	101.85	81.42	24.07	1.01	0	0.99	0.99	0	17.04	5.64	0	
19	112.76	102.08	81.19	24.07	1.02	0.01	0.99	0.99	0	17.04	5.64	0	
20	113.14	102.23	81.19	24.3	1.02	-0.01	0.99	0.99	0	17.04	5.64	0	
21	112.68	102.04	81.56	24.21	1.02	-0.02	0.99	0.99	0	17.04	5.64	0	
22	112.3	101.85	81.47	24.11	1	-0.06	0.99	0.99	0	17.04	5.64	0	
23	112.4	101.9	81.33	23.83	1.02	-0.01	0.99	0.99	0	17.04	5.64	0	
24	111.84	101.94	81.01	24.21	1.02	-0.02	0.99	0.99	0	17.04	5.64	0	

Figure 4.5 The measurement data from the data acquisition computer

This database will provide information such as

- *when will parts of an equipment should be replaced,*
- *when should an equipment be cleaned,*
- *what is the cost of cleaning,*
- *what is the cost of replacement*
- *whether parts of the equipment exists in the storage*

The whole data-handling module is object-oriented. An object-oriented database, on one side, will provide simplicity for the programmer to use the information and to expand the database, but on the other side, it is difficult for the non-programmer to fill in the database. The database is not compatible with common database applications such as MS Access. It is possible to convert this information into other database format but it must be programmatically[6] done. In short, this database lose its flexibility because its object-oriented structure. Chapter 6 will discuss this matter further.

4.3.2 THE BAYESIAN NETWORK MODULE

As will be explained, an additional Bayesian Network Module is developed to

a. Be able to use SmileX
b. Be able to communicate with process- and other data.

a) Rebuilding SmileX

The Bayesian network that has already modeled with GeNIe must now be loaded into our simulation. In order to do this we are using the ActiveX component called SmileX. This component still has many limitations such as the inability to retrieve the network structure; it can only retrieve its nodes and their value and do the basic operations such as updating the values and providing evidences into the network.

SmileX is based on OLE (Object Linking and Embedding) technology, a technology that provides a way to manipulate objects defined by an application or library from outside the application. The OLE standards are more difficult to program due to variable type conversions.

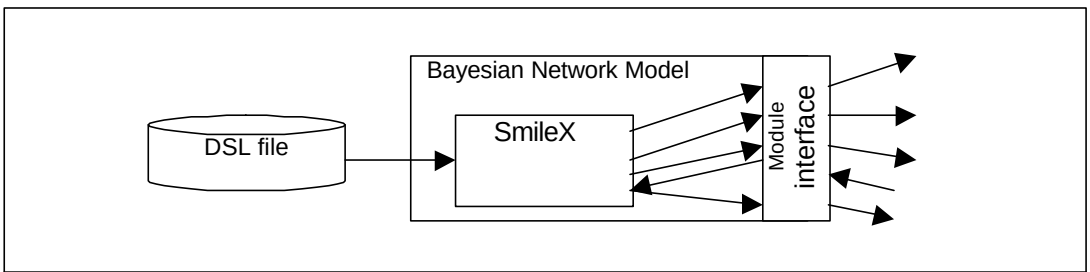


Figure 4.6 The enhancement of SmileX to adapt the simulation

[6] This conversion can be done through the standard Delphi component that allows us to read or write to an MS Access file. This components are the ADO components (ADO=ActiveX Data Object).

To illustrate the difficulties of OLE programming in Delphi, let us see the following example. In order to get the outcome of the Bayesian network read by SmileX object, we must write a conversion function from OleVariant into a well-known object in Delphi TStringList (this object is a list of string and very easy to use in Delphi). A conversion procedure would look like this:

```
procedure TForm1.Ole2StringList(o : OleVariant; var tl : TStringList);
var i : integer;
    b : boolean;
    s : string;
begin
   b := true;
   i := 0;
   tl.Clear;
   while b do
   begin
     try
       s := o[i];
       tl.Add(s);
       i := i + 1;
     except
       b := false;
     end;
   end;
end;
```

To retrieve the states of a nodes name we have to run this lines:

```
var o : OleVariant;
    Outcomes : TStringList;

   o := SmileX1.GetOutcomes(Nodes.Strings[i]);
   Ole2StringList(o, Outcomes);
```

This was only to retrieve the names of the nodes, to retrieve the value we have to run this line:

```
r := SmileX1.GetBeliefForOutcome('OK', 'Temperature');
```

The disadvantages of this method are:

1. We keep using at least two kinds of variables, the local variable that is directly accessible in Delphi (such as string, real, integer) and the 'universal' variable of OLE.
2. Conversion will always take place.
3. Due to the conversions and indirect use of the value of the variables, the structure of the program code is more difficult to read, especially for others.

The solution to these problems is to specialize the SmileX into a Delphi object module that communicates better with the Delphi programming language and customize the SmileX such that it can easily be applied to our Simula database. In other words we have built a new module that represents the Bayesian network but it still has SmileX in its core. This module gives a better interface to the programmer. Figure 4.6 shows us how the new module hides the SmileX module.

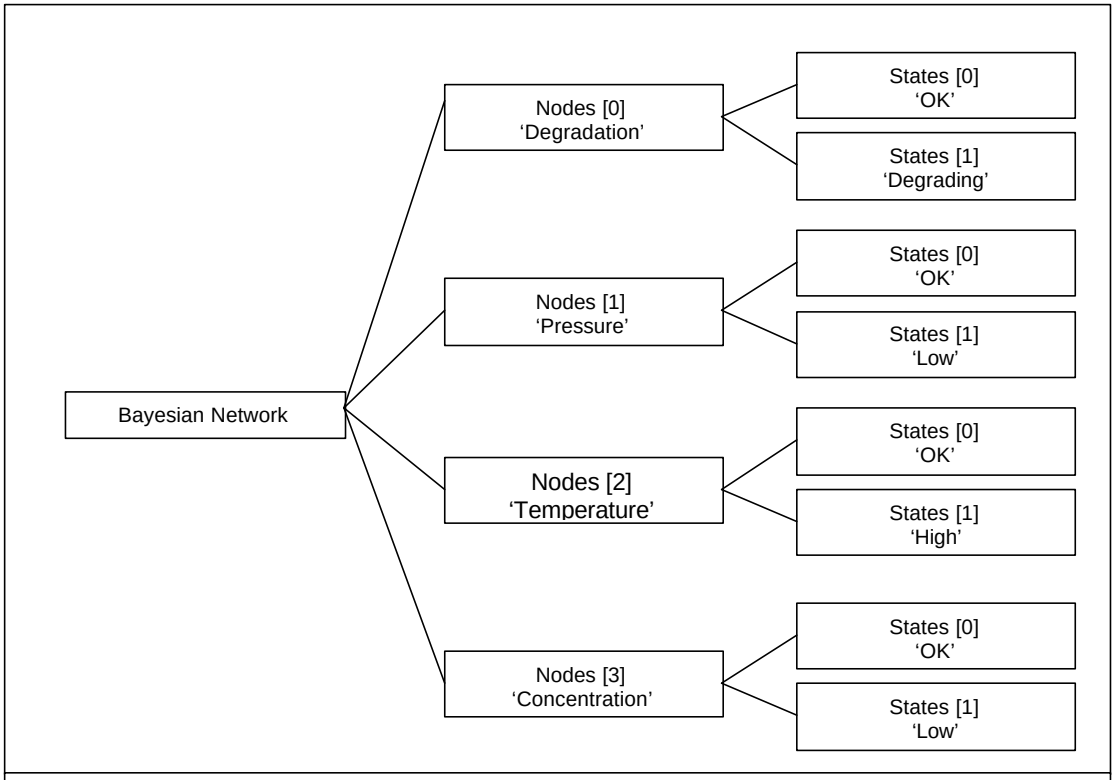


Figure 4.7 The conversion of the Bayesian network into a tree structure

With this new module retrieving the nodes of the Bayesian network are *virtually* saved as an array. Virtual means that it is not just an array it is actually a dynamic array. Since the user of the module sees the nodes as an array of nodes, no conversions will be seen and retrieving the node's properties can happen anytime and anywhere in the program code (see figure 4.7).

For example we want to retrieve the value of the state 'OK' in the node named 'Temperature'. Somewhere in the code we have a table that refers 'Temperature' to an index 2 and state 'OK' to 0. Then our program code would look like this:

```
Result := SmileX1.Nodes[2].States[0].Value;
```

The variable SmileX1 has an array called **Nodes** and each element of that array has also an array called **States**. Each element of **States** has the property called **Value**. That is the value of the state OK in the node Temperature.

This Bayesian network module hides the original SmileX component but the original functions of SmileX will still be usable.

b) Communication with the Process Data

The Bayesian network works with probabilities; no measurement values are used in the network. An expression like *'the temperature is 100℃'* would have no meaning in the Bayesian network. The language of the Bayesian network is such as *'the temperature is **OK** with probability 89%'* or *'the pressure is **High** with probability 20%'*. In the last section we discussed the data handling. The data that we use in this simulation are process-data, in the form of the first expression, not the second. Since the two kinds of data are not comparable, we have to define first what is ***High*** for a temperature. This Conversion module provides also that the conversion from process data into information in the Bayesian network terms. Figure 4.8 shows the translation scheme.

This conversion module allows us to save the information of the conversion rules. This function is useful when we want to enter a piece of evidence into the Bayesian network. With the help of this new function, entering a piece of evidence would be very simple, for example:

```
val := GetMeasurementValue;
k   := SmileX1.Nodes[2].GetStatusIndex(val);
SmileX1.Nodes[2].MakeEvidence(k);
```

The variable **val** holds the value of the measurement, and the method **GetStatusIndex(val)** gets the index of the state which will be activated according to the reference rule *if* the value on the variable **val** is entered. The second command set the evidence of node with index number 2, to the state with the state number that is given by the former command (that held by the variable **k**).

Concluding this section we can summarize that this Bayesian network module is an enhanced module of SmileX. There were two basic enhancements made:

a. That this module can be more easily used in Delphi programming and
b. Functionality has been added to compute the evidence more easily.

These two additional functions are made to customize to our Simula application. They are not fit for more general purposes (developing other general application based on

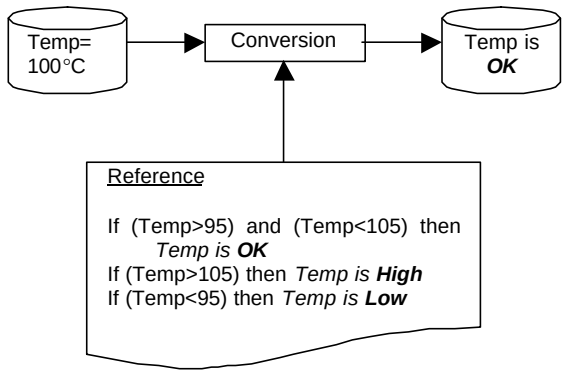


Figure 4.8 Translation flow from measurement data into a Bayesian variable value

Bayesian network).

The rule for each node state is in the form of defining in which interval is the state categorized. In this case we can use three kinds of intervals: a left open interval, closed interval and right open interval.

As we can see in figure 4.9 the first interval represents the left open interval, in the middle the closed interval and the last the right open interval. The value in the middle of the interval, 100, is called the fixed point and originated from the steady state simulation. This value is used as a reference, meaning that it is the ideal value.

To simplify we will represent this rule with two values and one sign. We will call the first value a reference value **Ref** that is the shift from the fixed point. The second value is called the interval value **Interv**, this is the width of the intervals. This second value is only used in the closed interval. To distinguish between the three kinds of interval we will use the sign **<, =,** and **>,** for left open, closed and right open interval respectively. For the open interval, **Interv** value would have no meaning since its width is infinite.

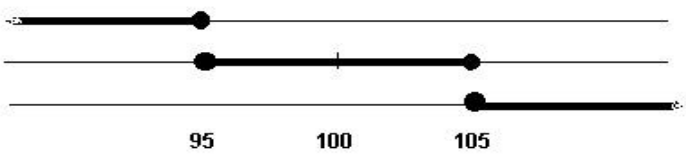


Figure 4.9 The mapping of the measurement data into Bayes' variable

In our example above our fixed value will be 100. To represent a **left open interval** from negative infinite to 95 we will use the following procedure. Since our fixed value is 100, next we must shift this value 5 to the left, which means that our **Ref** value is –5. The sign to be used here is '**<**'.

To represent **closed interval** from 95 to 105, the middle of this interval is 100 that is exactly the same as the fixed point, then the shift is none, **Ref** value is 0. Then the width of the interval is 5 points to left and 5 points to right, the **Interv** value is in this case 5. The symbol that will be used here is '='.

To represent **right open interval** from 105 to infinity, we will take a shift of 5 points to the right of the fixed point, the **Ref** value is 5. The sign used in this representation is '>'.

For defining the rule for the conversion a graphic user interface is made. This user interface is as shown in figure 4.10. On the left side of the dialog box is the list of the nodes of the Bayesian network. On the right side are the parameters of that node. If we want to define a node as input for the Bayesian network we can check the *check box* ***As Input***. Then we must choose to which equipment and to which point it corresponds. These selections can be chosen from the two *combo boxes* beneath the *fixed value*. Just beneath the combo boxes we can define the rule for the each state.

For each state we can choose three signs '<', '=', or '>'. Then we can fill in the **Ref** value and the **Interv** value. It is very important to define each state of a node that used as input. Each node used as input will be boxed with red lines. Nodes, boxed with blue lines, are used as output but it will be discussed in chapter 7.

4.3.3 RULE-BASED ADVISING MODULE

This module is the last stage of the simulation. This module gets information from the Bayesian network. Based on this information and predefined rules, this module will provide advice to the user whether to revamp the whole plant, or to replace parts of the equipment.

There are no special tools to build this module. This module consists of if-then-else rules that are hard coded in the application. It means that once the rules are programmed, it cannot be changed easily. To change the rule, we have to change the program code.

This module receives information such as *the degradation of the Reboiler* is *5%*. Each peace of equipment's degradation is simulated in the Bayesian network, and the rule-based advising module uses the result. By applying rules, this module will compute the most profitable decision. This module will also use the information of the cleaning cost and replacement cost for each piece of equipment for the calculation. To show the result to the user, this module is connected to a graphical user interface.

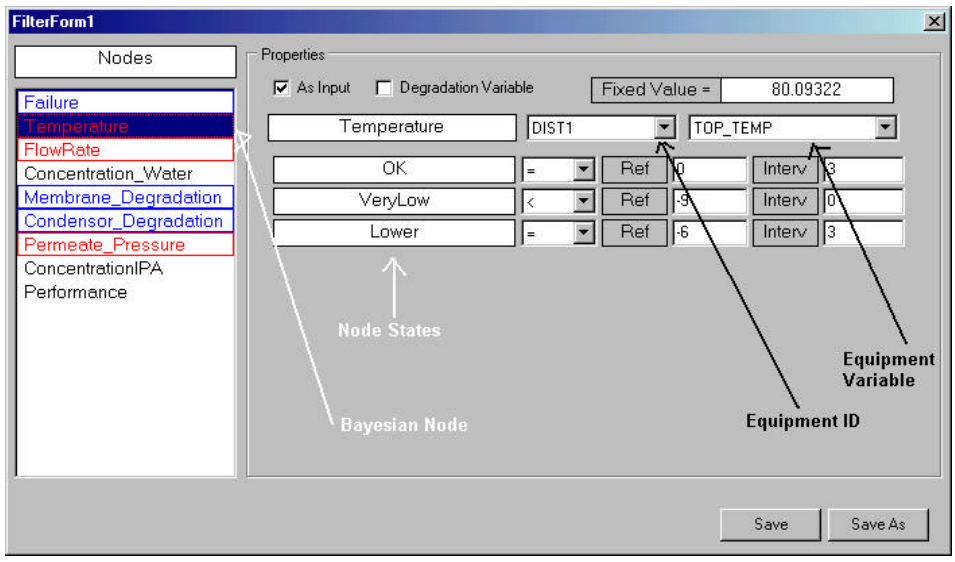


Figure 4.10 the user interface of the application to set the conversion to the Bayes' value

4.4. DEVELOPING THE APPLICATION

4.4.1 PROGRAMMING LANGUAGE

This application was written in Delphi (the successor of Turbo Pascal) and runs under Windows 98. Delphi is a product of Borland Inprise; it is a company specialized in development software. Delphi is one of its main products. Delphi is based on object-oriented Pascal. Pascal itself is a high level programming language (programming language which uses more friendly terms and easier to understand because its similarity with natural language). Right now Borland has released Delphi 5.0, which has been used to develop this simulation.

Several reasons can be mentioned why we used Delphi in developing this application:

1. Delphi is a high level programming language, which has the abilities comparable to C++. With a high level programming language an application can be developed faster than when it is written in a lower level programming language. Of course it depends as well on the familiarity of the programmer. In this case Delphi has the most suitable interface for the programmer.
2. Delphi supports object oriented programming. This makes it possible to program an object-oriented database.
3. Delphi supports the integration with ActiveX components. In order to use the SmileX component, it is important that the compiler has the ability to load ActiveX components.
4. Delphi offers additional components. These components are saved in a standard Delphi library that is compiled within the executable file. Unlike compilers from Microsoft such as Visual C++ and Visual Basic, where the components are saved in several library-files instead of being integrated in the executable file. Therefore, with Delphi it is easier to transport the program to other computers since the components are built in the executable file itself. In Visual C++ or Visual Basic, the library files must be transported with the executable file. It is not always comfortable for the developer to transport those components to other computers.

This application is developed and tested on a PC with 200MHz-AMD processor and 32MB memory. It runs a little bit slow, thus a faster PC and a bigger memory are recommended to run this application. It is assumed that this PC configuration is the minimum configuration to run this application.

4.4.2 PLATFORM

This simulation is developed available under Windows 98. Considering that Windows 98 is the most common platform used in this area. Another reason why it is developed under Windows is that Delphi is also under Windows. There are not yet other development tools that seem to be as powerful as Delphi and that are running under other platforms. In fact, Delphi has recently ported parts to Unix. Since Windows is considered good enough to run of this application, there is no need to use a machine-independent compiler such as Java.

5 BAYES AND THE LIFE CYCLE SIMULATION

5.1 INTRODUCTION

The Bayesian network model is very important for this Life-Cycle simulation. The model is applied to analyze the behavior of the elements of the process including the prediction of the result or prediction of degradation of some equipment. This network is built as a causal inference model based on prior knowledge. Therefore, the network should be predefined before it can simulate the process.

In defining the network there are a few steps to follow:
1. Understanding how the process works.
2. Analyzing which outcome we want to predict (degradation, performance, etc).
3. Analyzing which variables are important in the process.
4. Analyzing the causal connections between the variables
5. Constructing the network

These five steps will be discussed in the following sections.

5.2 PARAMETERS OF THE PROCESS

The process we want to approach is a hybrid of distillation and vapor permeation process through membranes. To find the important variables it is important to understand about the process itself and how it works. We can consider this process as two processes, the distillation and the vapor permeation. We will discuss these two processes separately.

5.2.1 THE DISTILLATION PROCESS

Process Description

The idea of the distillation process is to separate a mixture that has different boiling temperatures. The most important part of the distillation process is the *distillation tower/column*. The mixture enters the tower at a specific height and the temperature in the column is heated by a *reboiler*. The reboiler gets the energy from steam. The reboiler must heat the column to a specific temperature where one component of the mixture is just vaporized and the other component just about to vaporize. The component with higher boiling point will go to the bottom of the column and the other component, which boils faster than the other, will go to the top of the column as vapor (see figure 5.1).

At the top of the column the vapor is condensed by a *condenser*. This condenser makes the temperature of the vapor lower, so low that the vapor condenses into liquid. We call this stream the *distillate*. This liquid may not reach the optimum purity at the first stage at the first configuration, and that is why there are several stages that occur in the column.

At each stage applies a mass balance:

$$F_n + V_{n+1} + L_{n-1} = V_n + L_n$$

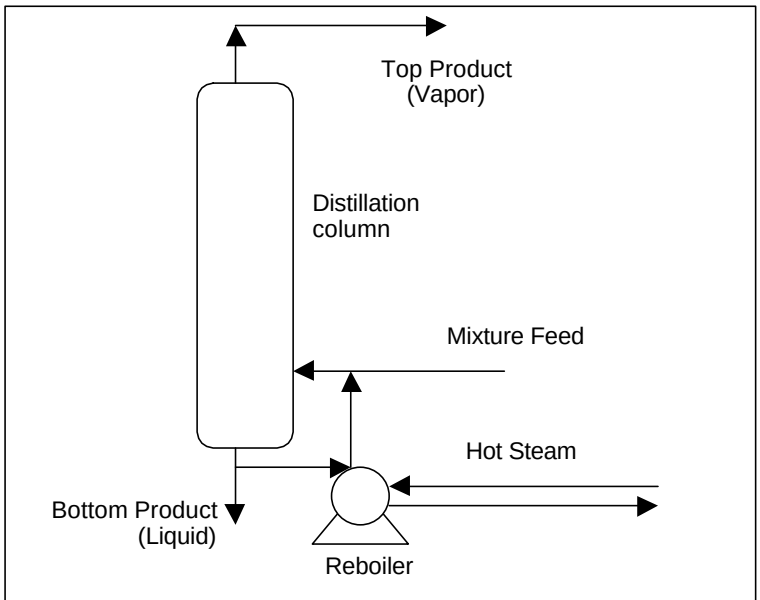


Figure 5.1 Scheme of a basic distillation process.

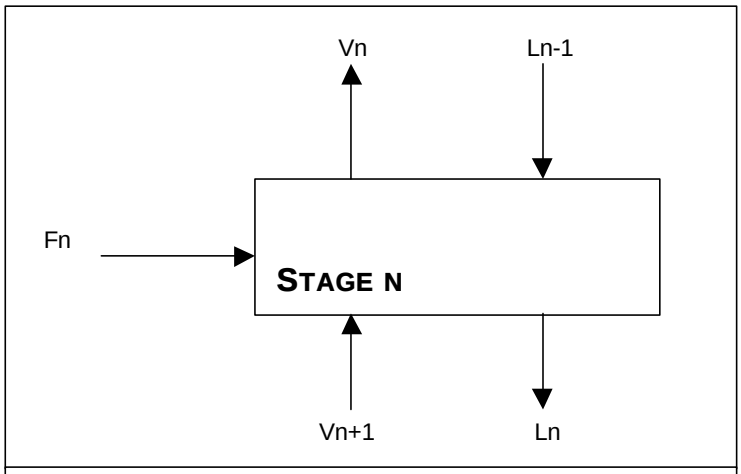


Figure 5.2 The mass balance scheme in a stage

$F = feed$

$V = Vapor$

$L = Liquid$

$n = stage\_number$

The vapor from the lower stage is called V_{n+1} and vapor to the higher stage is called V_n. Th produced liquid goes to the lower stage this is called L_n and incoming liquid from higher stage is called L_{n-1}. Each stage has its own process conditions like temperature, mass fractions for each component (vapor produced, vapor leaving the stage, liquid produced and incoming liquid). These conditions are calculated in the steady state simulation.
Using multiple stages is meant to improve the separation results. Besides multiplying the stages another techniques are also used, *stripping, rectification and fractionation stages*. In the stripping stage the liquid product at the lowest stage (highest stage number) is heated so that this liquid vaporizes and reenters the column.
In the rectification is just the at topside. A portion of the vapor product at the highest stage (lowest stage number) is condensed and it reenters the column. This liquid recycle is termed *reflux*. A combination of stripping and rectification is called *fractionation* (see figure 5.3).

Details of the process i.e. the equations to compute the process outcome are not discussed in this thesis because its not the main goal of this thesis. We assume that the computation of the process is known and can be calculated by simulation programs such as ASPEN.

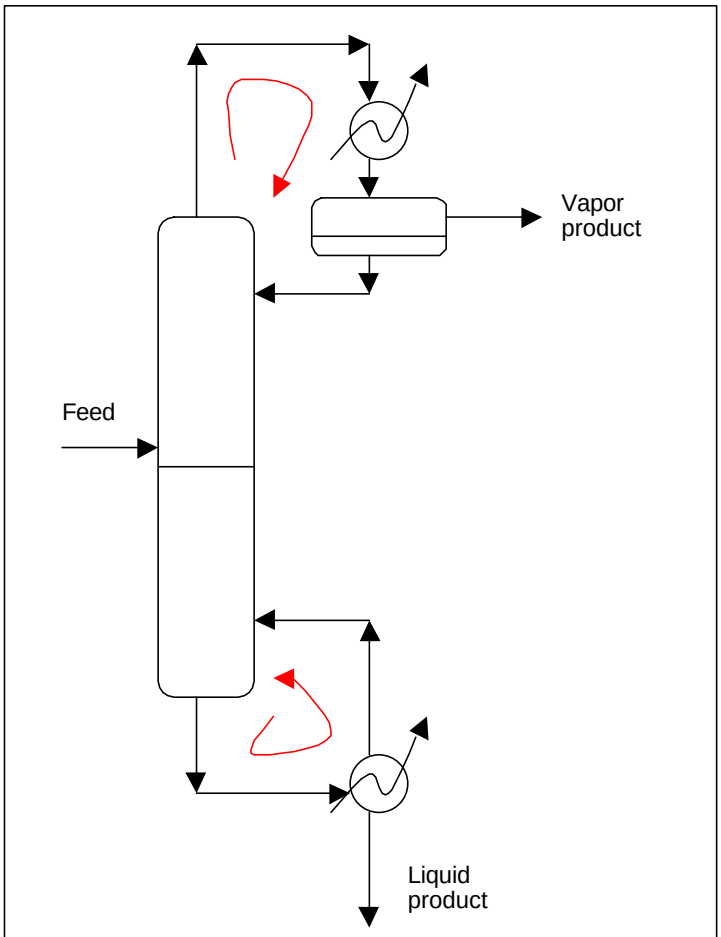


Figure 5.3 Fractionator scheme

Analyzing the Process

As described in the previous section, there are three pieces of equipment, which are important in the distillation process. These three pieces of equipment are *distillation column, reboiler and condenser*.

The distillation column is the place where the process takes place. The main function is to make an isolate system so that the process can maintain the process condition such as particular pressure and temperature. It does not mean that the column controls the condition, but it just makes it possible to maintain the conditions. Degradation of this piece of equipment is very low and can therefore be ignored.

The reboiler is a heat exchanger. It gives heat to the mixture so that the mixture can vaporize. The reboiler plays an important role in distillation process because the process should be maintained at a particular temperature. The temperature should not be higher than the highest boiling point of one of the component, and it should not be lower than the lowest boiling point otherwise it would not boil anyway. Degradation impacts its functioning and subsequent outcome of the process.

The condenser plays also an important role in this process. It cools off the vapor. A lower temperature means a lower pressure too. To raise the column pressure, the flow

of condensate from the condenser is lessened. This increases the flooded area in the condenser and lowers the surface area exposed for vapor condensation. This in turn reduces condensation rate, thereby raising pressure.

Besides, the condenser controls the amount of vapor which has to be condensed and reentered the column. It controls the *reflux ratio*. This reflux ratio has an influence to the distillate purity. A condenser is actually a heat exchanger. It cools off vapor by mean of letting the vapor through a cool element, just like a car radiator.

Some consideration for the condenser:
1. If the condensing temperature is high compared to ambient and the condensing range is narrow, condensation on the reflux drum walls may interfere with the condensation or pressure control. Insulating the vapor space of the reflux drum should be considered.
2. Streams should enter the reflux drum at a velocity low enough to prevent disturbance to the liquid surface.

The degradation of a condenser itself may be considered low, but there can be a few disturbances for the condenser so that the pressure in the column is not as it should be.

The main function of a reboiler is to produce an amount of vapor from the mixture feed. A direct consequence when the reboiler is not working properly is that the temperature at the bottom of the column is not as it should be. A kind of disturbance that can happen in the reboiler is the contamination of the reboiler. A contamination in the reboiler can make the heat exchange ineffective.

Process Parameters
As the previous section described, we are interested in the degradation of the pieces of the equipment. The most important pieces of equipment in distillation process in which its degradation is essential for the process are the reboiler and the condenser, since these pieces of equipment influence to the temperature and pressure.

From the analysis above we can deduct the following knowledge rules with regards to data:

Reboiler
- If degrades then the bottom column *temperature* lowered.
- If the temperature becomes lower then the amount of vapor becomes less.
- If the vapor lessen then *distillate flow rate* will lessen.

Condenser
- If there is a disturbance then it cools off less vapor and contains more vapor than wished for. The temperature becomes higher also.
- If more vapor produced then reflux will lessen.
- If less reflux is made then the process conditions will be disturbed which means the product purity is not as predicted.

This clarifies which parameters are to be used in our model. And these are the parameters that will be used in the model (see table 5.5 and 5.6).

Table 5.5 Parameters for a reboiler

Output:	Reboiler degradation
Input:	Column bottom temperature
	Distillate flow rate
	Concentration in Distillate

Table 5.6 Parameters of a condenser

Output:	Condenser degradation
Input:	Column Pressure

5.2.2 THE VAPOR PERMEATION PROCESS

Analyzing The Process

The idea of this process is to let one component of a mixture through membranes due to the differences in solution and diffusion of compounds in the membranes. Characteristics for the membrane unit are its selectivity or normalized flux (flux divided by driving force). When the mixtures are in the form of gas or vapor then the

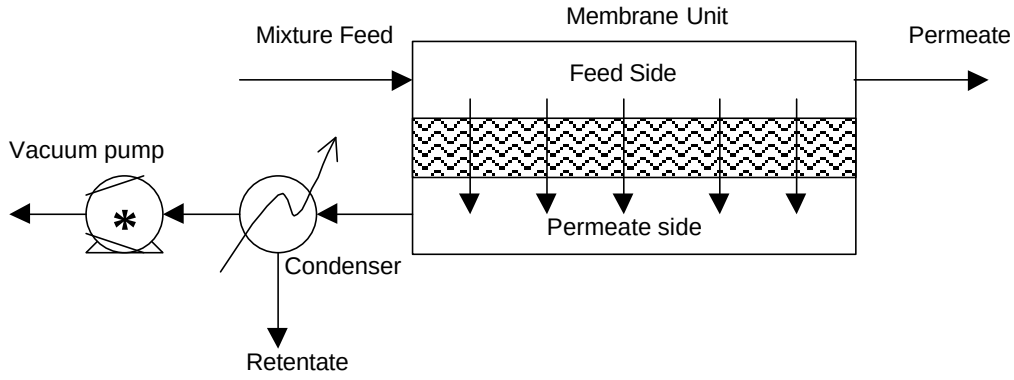


Figure 5.7 A schematic view of the vapor permeation process with membranes

normalized flux is called permeability coefficient. Flux itself is indicative of the flow rate (of permeate) through the membrane. The flux is expressed in mole per meter square second ($^{mole}/_{m^2.s}$). The driving force is the force that makes the vapor flows through the membranes, in this project the *permeate pressure* (that is lower than the pressure in the membrane unit, due to the vacuum pump) outside the membrane unit is used as the driving force.

Figure 5.7 let us see the schematic view of the vapor permeation in general. The mixture in the form of vapor enters the membrane unit. The selected component of the mixture will be passed through the membranes to the permeate side. The component that is not selected by the membranes passes as *permeate* (the product from the retentate side). In this case the permeate will be water rich, because the IPA will be let

through the membranes. The IPA will be condensed by the condenser and transferred to the outlet as *retentate* (product of the permeate side).

In the pilot plant report, the following results are reported:

- The membrane module has a high selectivity in low permeate pressure. This can be explained by the results.
- The water mass fraction in permeate is high in low permeate pressure and high flow.
- The IPA mass fraction in retentate is high in low permeate pressure and low flow rate.
- The temperature has some influences on IPA/water mass fraction in retentate and permeates in this new module.

Process Parameters

It is clear, when we look at the results, that there are a few parameters that are important in this process. Since our product is the IPA, the third point of the above result may be the key to our model. The concentration of the IPA is strongly influenced by the permeate pressure, that is the driving force, and the flow rate. A higher permeate pressure (controlled by the condenser) may cause a lower performance, an optimum performance can then be obtained in an optimum permeate pressure (not too high). Also a higher flow rate can lead to a membrane that is not working effectively, and therefore an optimum flow rate should be maintained.

We can now conclude that the important parameters for this process are shown in table 5.8 and table 5.9

Table 5.8

Output:	Membrane degradation
Input:	Distillate flow rate
	Permeate pressure
	Concentration in Permeate

Table 5.9

Output	Condenser degradation
Input:	Permeate pressure

5.3 CONSTRUCTING THE MODEL

5.3.1 THE CAUSAL RELATIONSHIP

Until now we have gathered the key parameters of the hybrid distillation/ vapor permeation process. These parameters will be used to construct our Bayesian model. Before we build our network, let us take a look at each parameter and put them into a table. In this table a score is added in each influence. This score indicates how strong the influence of *cause* and *effect will be*. For simplicity we will use the scale 1 to 3.

The score is acquired with the help of an expert in chemical process, by means of interviews.

Score:
[ ] 1 = a little influence
[ ] 2 = moderate
[ ] 3 = sensitive

Table 5.10 The sensitivity of the influences among the variables

Influence \ Effect	RD	BT	BW	P	DF	RC	MD	CD	PP
		2			3				
BT			3						
BW				2					
P									
DF				2		1			
RC				3					
MD						3			
CD									1
PP						3			

RD Reboiler Degradation
BT Bottom Column Temperature
BW Bottom Water Concentration
P Performance
DF Distillate Flow Rate
RC Retentate Concentration
MD Membrane Degradation
CD Condenser Degradation
PP Permeate Pressure

We can formulate table 5.10 in another way:

Table 5.11 The influence of the process variables to the another variable

No	Parameter	Influence on	Score
1	Reboiler degradation	Bottom Column Temperature	2
2	Bottom Column Temperature	Bottom Water Concentration	3
3	Bottom Water Concentration	Performance (Quality)	2
4	Reboiler degradation	Distillate Flow Rate	3
5	Distillate Flow Rate	IPA Concentration at retentate	1
6	Membrane Degradation	IPA Concentration at retentate	3
7	Condenser Degradation	Permeate Pressure	1
8	Permeate Pressure	IPA Concentration at retentate	3
9	IPA Concentration at retentate	Performance (Quality)	3
10	Distillate Flow Rate	Performance (Quality)	2

To summarize this we can formulate these following knowledge rules:

1. Reboiler degradation → Bottom Column Temperature
   When the reboiler degrades (contaminated) the heat transfer is not effective. This causes the bottom column temperature become less. This is a direct causal relationship with an average score.

2. Bottom Column Temperature → Bottom Water concentration
   When the bottom temperature becomes less it disturbs the distillation process and changes the steady state of the process. When the bottom column temperature drops there will be less IPA vaporized, and therefore the water concentration at the bottom of the column becomes less as well.

3. Bottom Water Concentration → Performance
   One of the measurements for the performance is the water concentration at the bottom of the column. It should be close to the required purity.

4. Reboiler Degradation → Distillate Flow Rate
   When the reboiler contaminated there will be less vapor produced and therefore the vapor flow rate at the top of the column will be lower.

5. Distillate flow rate → IPA Concentration at retentate
   As the research concluded that as the flow rate become higher it will disturb the concentration of IPA in the retentate.

6. Membrane degradation → IPA Concentration at retentate
   The result at the retentate is very sensitive in the defect of the membranes. This information is confirmed by experimental results.

7. Condenser Degradation → Permeate Pressure
   When the condenser is not working properly then the permeate pressure will become higher, since the condenser is not able to lower the temperature as it should do.

8. Permeate Pressure → IPA Concentration at retentate
   The vapor permeation is as well very sensitive also in the retentate pressure since this pressure provides the driving force for the vapor. When the pressure is higher, then the selectivity will become lower.

9. IPA Concentration → Performance
   This IPA Concentration has a direct influence on the performance since this is the main product of the whole process.

10. Distillate Flow Rate → Performance
    When the distillate flow rate is lower it will not lessen the IPA concentration at the retentate but a lower distillate flow rate means less quantity IPA produced. That is why this parameter has as well an influence on the performance.

We have now a list of influences. With this list we can now build the Bayesian network model as illustrated in figure 5.12.

5.3.2 THE NETWORK MODEL

Using the list of the influences we can now construct our network model. The variables of the process can be translated as a node in the graph and the influences can be translated as the vertices in the graph. This model can then constructed with the tool application GeNIe. Figure 5.12 shows us the complete Bayesian network model for the hybrid distillation and vapor-permeation process.

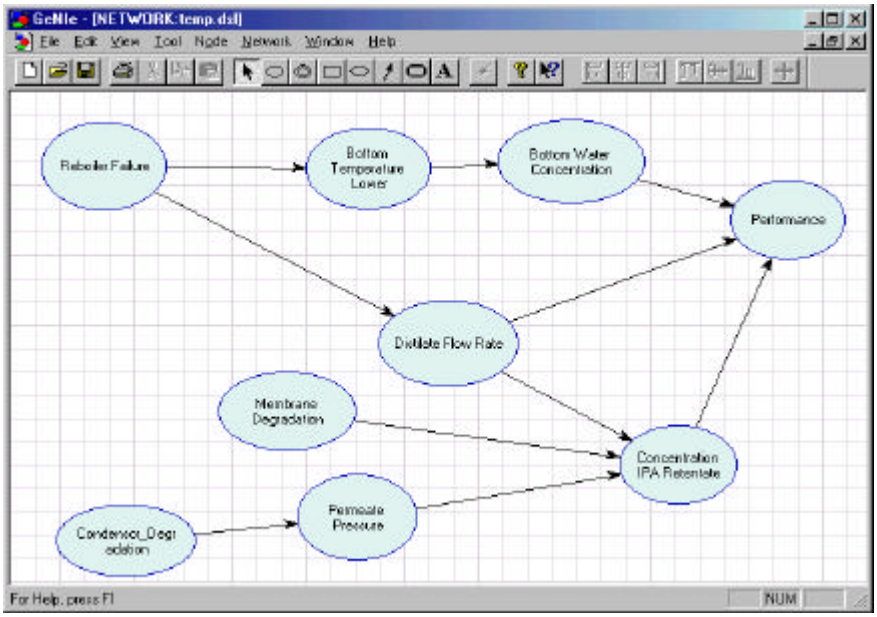


Figure 5.12 The model for distillation/vapor permeation process

The definition of each node is based on prior knowledge of the causal relationship. It is based on the scores that have been filled in the table of the influences. The exact probabilities of the influence are fine-tuned by a chemical engineer, the expert, and later, as well validated by him.

5.4 MODEL VALIDATION

This model is validated by means of analysis, expert review and data validation. The first step of the validation consists of analyzing the process. Using literatures that explain the process the designer will get the first impression of the process itself and can select some important key parameters that he will use in the model. Except from literature the designer has been guided by a chemical engineer, the expert, to understand the process.

In the second step, a rough model of the process is built. This model can easily be examined by someone that has no experience with the Bayesian network. In this case a chemical engineer can intuitively examine the causal relationship in the Bayesian network. A few changes can be made concerning the reasoning the causal relationship.

The third step is fine-tuning the process. By entering the matching data into the model and see if the probabilities match the data, we are fine-tuning this model. In this model we were using the experiment data from the past. There are several kinds of data sets and also with several reference data. And as at last step, we will ask for an expert review to examine whether this model is working properly.

Some parts of the model have been changed due to the causal ordering of the events. Experts can easily see these kinds of errors because the graph can be intuitively read. Examples of these kinds of errors are:

- *"The raising of distillate flow causes the increasing IPA concentrate in the retentate"*, where it should be "*the raising of distillate flow causes the decreasing IPA concentrate in the retentate"*
- *"The retentate concentration is very sensitive to the distillate flow rate"*, which should be, "*the retentate concentration **is** influenced by distillate flow rate, but the influence is not great, because the retentate concentration is more controlled by the permeate pressure"*

A few minor changes like how much an event influences another have also been made by letting experts run the model. The fine-tuning of this model is done together with the expert.

The model built for this simulation is constructed and validated according to above steps with the help of some chemical engineering experts at the Laboratory for Process Equipment at the Delft University of Technology.

5.5 APPLYING THE MODEL

To apply this model we should first define which nodes we are going to use as input, and which nodes as output. In the Bayesian network model we have the freedom to use each node as input. Input for Bayesian network is in the form of evidence. Therefore the input for our Bayesian network model depends on the measurement instrumentation of the process. The more measurements that match the node of the network the more accurate will the network predicts.

In the pilot project as performed in the Laboratory of Process Equipment in Delft, in-line instrumentation has been used for measuring temperature, pressure and flow rate. The flow rate is either in volume or in mass. No online measurement was available for concentration. However, this could be done off-line in the lab. For our model we can use the nodes that represents the temperature, pressure and flow rate as input for our model. In the application the nodes and its prior probabilities can be displayed. The structure of the nodes cannot be retrieved from the file format from GeNIe. This is one of the disadvantages when using SmileX.

The data set used in this model comes from the data acquisition program called TestPoint. This data is measured periodically and saved to a file. This is a file with a large amount data. Each record of this file is a set of measurement at the same time and each of these records is used as input in our Bayesian network model. The measurement is translated into evidence for the network. After all input is entered, the application will notice also the outcome of the other nodes. Figure 5.13 shows all the nodes as they are listed from the left to the right with all its states. For each record its data are filled into the network and the result is put in the grid.

When the sequence is completed for all records then the outcome will be averaged and displayed.

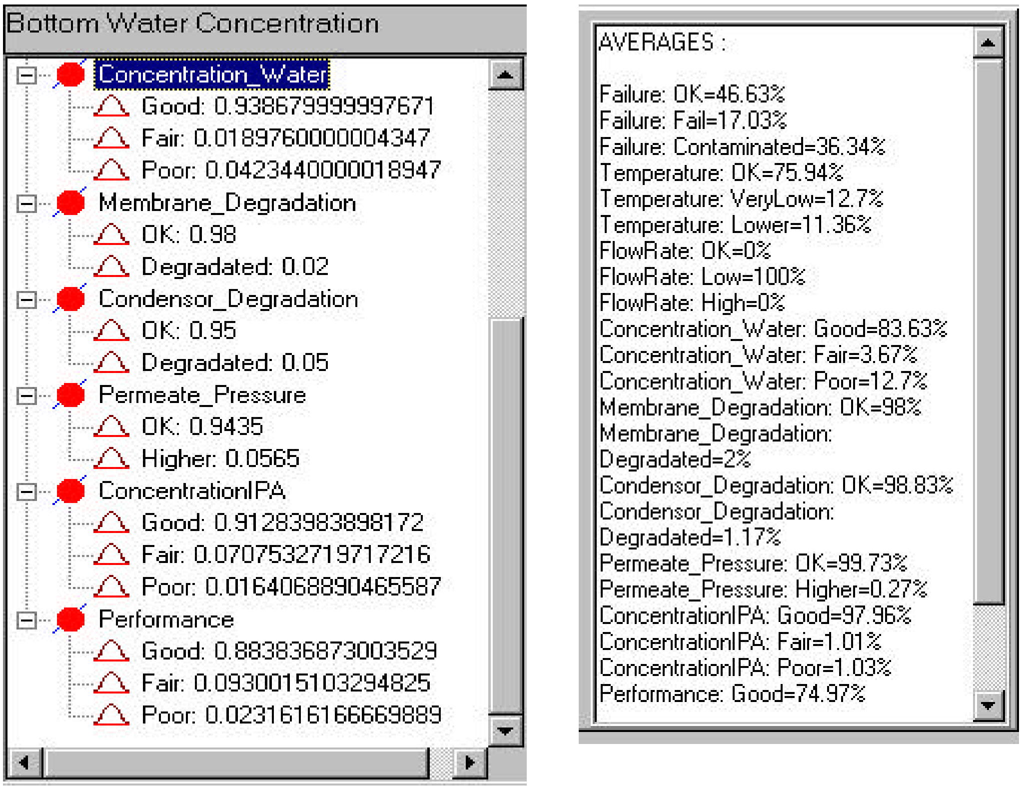
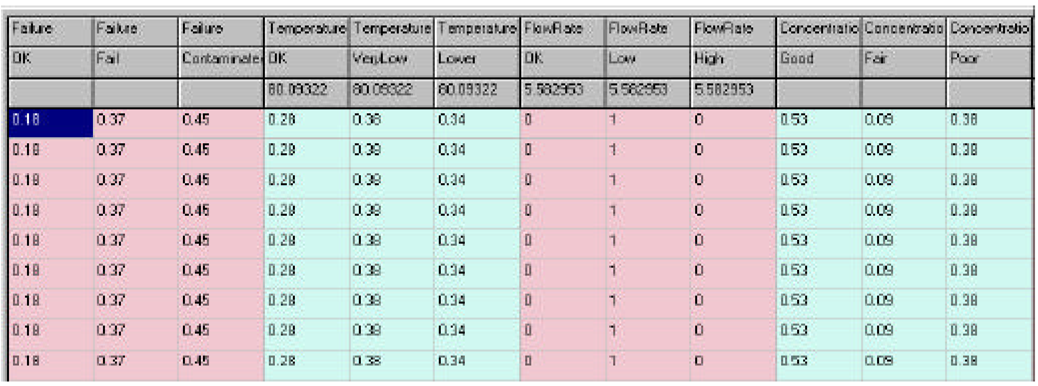


Figure 5.13 Nodes of the model and its prior probabilities and results of calculation

The outputs of our Bayesian network are in this case *the degradation of each piece of equipment* and *the performance of the process*. The degradation of each piece of equipment is used to define the total degradation and later to decide the revamp action. The performance is also used to measure the revenue. When the performance is low, the process will also deliver less revenue.



Failure	Failure	Failure	Temperature	Temperature	Temperature	FlowRate	FlowRate	FlowRate	Concentratio	Concentratio	Concentratio
OK	Fail	Contaminate	OK	VeryLow	Lower	OK	Low	High	Good	Fair	Poor
			80.09322	80.09322	80.09322	5.582953	5.582953	5.582953			
0.18	0.37	0.45	0.28	0.38	0.34	0	1	0	0.53	0.09	0.38
0.18	0.37	0.45	0.28	0.39	0.34	0	1	0	0.53	0.09	0.38
0.18	0.37	0.45	0.28	0.39	0.34	0	1	0	0.53	0.09	0.38
0.18	0.37	0.45	0.28	0.39	0.34	0	1	0	0.53	0.09	0.38
0.18	0.37	0.45	0.28	0.39	0.34	0	1	0	0.53	0.09	0.38
0.18	0.37	0.45	0.28	0.39	0.34	0	1	0	0.53	0.09	0.38
0.18	0.37	0.45	0.28	0.39	0.34	0	1	0	0.53	0.09	0.38
0.18	0.37	0.45	0.28	0.39	0.34	0	1	0	0.53	0.09	0.38
0.18	0.37	0.45	0.28	0.38	0.34	0	1	0	0.53	0.09	0.38

Figure 5.14 The simulation result from the given the measurement data

5.6 CONCLUDING REMARKS

Our model was mostly built with a help of a few experts in process engineering. The role of the process engineers is to validate the model. As has been written in the previous sections this model is based on analysis of how the process works and what are the important pieces of equipment from both process and economical point of view.

At last the model was integrated to the application and tested. At this phase the model was once again reviewed by the expert to fine-tune the model. When the model has been fine-tuned then it is ready to simulate other cases from the same process model. The model is specific for a process, it means that to simulate another process, new model has to be built, reviewed and fine-tuned.

6 THE DATABASE MODEL

6.1 INTRODUCTION

In this chapter we will discuss the selection and realization of the database model that is built for the Life Cycle simulation. The first question that might be asked is why building a new database structure. Since the incoming data can be stored in the specific structure and pulled up to perform the Bayesian calculations. There are several reasons to address this question. The reasons are partly practical – to solve a current problem - and partly based on a broader, more flexible concept that fits the Life Cycle paradigm – which requires future flexibility. This flexibility is as well related to the selection of the database as on the data modeling. Thus, these two aspects will be discussed. First, we will look at the selection of the database-approach; secondly, we will discuss the data modeling approach.

6.2 SELECTION OF DATABASE APPROACH

According to its purpose, a database can be classified as *a general-purpose database* and a *special-purpose database*. A general-purpose database is a generic relational database such as Oracle, SQL or MS Access and others. They should have a broad appeal and its application should provide a very wide functional support, can handle regular requirements and fit to deal with gigantic quantity of data and to process the data quickly. A special-purpose database has a narrow application. Usually, it is built for a special set of functions and is optimized for those functions. These may include speed and specific complex transactions, etc.

A general-purpose database has to fit generic sources of data for the administration of employees, clients, for financial and project data. A general-purpose database management system can handle regular requirements, like querying and handling the transactions. The source of data usually comes from management tools that can easily be converted into this general-purpose database.

A special-purpose database includes special conversions from other sources of data and therefore the source of data can have a very wide margin. The special-purpose database can include a wide degree of freedom for the modeling. The disadvantage is the need to implement solutions from the lowest level to the highest level. It means that the programmer / designer should implement the database from the file handling to the user interface.

The Life Cycle Simulation application receives data from a very large variety of data-sources. It brings technical specs and financial performance data together as well. The quantity of data used in this application can also be so extensive, especially with regards to in-line metering. These facts have brought several difficulties concerning the data:

- When using data from a few sources, the information in the database should be selected. Not all data in the database are useful for the simulation.
- The data from different sources use as well different formats, different units, etc. Different formats should be converted and different units should be converted too in order to compare them.

Another aspect is the nature of Life Cycle simulation. This thesis describes a limited illustration of the Simulation. In the real world, over the Life Cycle of a plant, different applications may change (a new ERP system, for example). This would require a set-up that provides flexibility to handle changes over the course of the Life Cycle. Therefore, one has to choose between either data handling from many sources that is based on *hard-coded programming* that is specifically built for each sources or provide a more flexible set-up. The first solution, a hard-coded program, loads more duties onto the simulation-program and may hamper future changes and additions. Therefore, a solution to this problem is to develop a “modular” approach that allows for the Life Cycle environment to add another module or modified it during its existence.

The solution to these difficulties is to build a specific database module. That module of the system – the database - should be based on the following requirements:

- Integration.
  To provide simulation integrated data from sources, such as process simulation, maintenance data and data from data acquisition systems.

- Standardization.
  Different sources imply different types and formats of data when actually some of the data is meant to be the same. This database will include functions that standardize the data formats and types.

- Uniformity.
  To enhance a uniform interface and system environment.

- Specific data entry Data Processing.
  To standardize data before storing in the database. This data entry processing includes technical, financial and maintenance data processing. It can be set-up in a real-time mode, thereby providing online conversions

6.3 OBJECT ORIENTED DATABASE

The second aspect to be discussed in relationship with the database was the need to provide a structure that will provide flexibility towards expansion and towards the future. The relational database provides a limited flexibility towards its implemented structure. It is difficult or impossible to add new main items with their characteristics and attributes. If switching the structure from one database to a "next" version or generation of a relational database, the hierarchy has to be ported as well. That will be a difficult changeover as well. Thus, with regards to flexibility, the relational database has deficiencies.

The technology to be able to do this in a flexible way is called "object orientation". This approach provides means of classification of objects that enhance integration of aspects around an object and at the same time defines a hierarchy is such a way that information on a higher level can be consistently linked to lower hierarchies. The integration is based on the definition of the structure of object classes and attributes. The hierarchical principle is based on inheritance.

The concept of object-oriented database is to assume the real system as objects. Each of these objects has its own attributes (properties) that make each object instance unique. Every object that has the same attributes is classified into the same object-class. Each element in that object-class is called object-instance.

This method of modeling a database is less complex since the model resembles the real system. For example let us take our plant in a more simple way, an object-class called **Plant**, this object-class has properties such as: *plant identity, location, what the products are, what the raw materials are,* etc. This plant can be viewed as a collection of processes. Each of these processes uses pieces of equipment.

The equipment has its unique properties like *equipment identity, to which plant it belongs, for which process it is needed, when it should be cleaned and how,* etc. These common properties of all kinds of equipment form a new object class called **Equipment**. There are several kinds of equipment and each kind of equipment has its own more specific properties, but basically they are all equipment. Here we are talking about *generalization* and *specialization*. The general object class for all equipment is **Equipment**, and it can be specialized to **Column, Reboiler, Condenser** or **Pump**. The specialized model of equipment *inherits* the properties of its basic model and the object class **Plant** can still view a **Column** as **Equipment**. When a special form of pump is used, e.g. vacuum pump, then the object class **Pump** can be once more specialized and object class **VacuumPump** is created. This new object class can still be treated as a normal object class **Equipment**.

Object-oriented database is more suitable when we want to treat object models as real objects and not just in the form of tables and queries. The most important thing is not only the relationship but also the object as a whole class apart from other classes. The database does not look like tables that are linked one to another but it looks like a bag containing object-instances and each instance is built up from other objects. This approach is more natural than the conventional database based on *entity relationship*.

Beside the inheritance, object-oriented databases have the so-called *method*. In the conventional 'programmers world' it is called *functions* and *procedures*. They are not just functions and procedures. These functions and procedures are used for information transactions in the object and for data processing in the object. Back in our object-class **Equipment**, it may contain *methods* like *need cleaning, clean equipment, read status*, etc. These *methods* are also inherited when specialization is made from this class and modification can be brought to method of the specialization when it is necessary.

This is as well helpful in defining and realizing specific functions such as modifying and integrating data. The modeling in an object-oriented database is quite natural. The structure of the database is nearly the same as the real object that we want to model. We have chosen an object-oriented database model to be used in our Life Cycle simulation application. This choice is based on the following reasons:

- The orientation of the simulation is more on objects than on the data. Measurement data may be gigantic but they are used only to compute the simulation; it is not necessary to take back a particular record from a particular timestamp.

- The variations of the set of equipment will be large. Each type of equipment shall have its own unique attributes. In object-oriented it is easy to make an inheritance from existing objects and modify or add some attributes. In conventional database this is not easy to do.

- Flexibility in the programming, i.e. for the conversions procedures and data manipulating in the program code. When conversion procedure has to be written it can be built in the object as part of the object itself. When objects are used in the programming, manipulating the data is easier than to connect to an external database. Easier means that the attributes of the object can directly be treated as variables in the program code rather than to specify first the table name, field name and record number.

In the future, this selection provides flexibility towards web-based enhancements of the system.

6.4 IMPLEMENTATION OF THE PROCESS DATABASE MODEL

In the Life Cycle simulation we are going to model the plant at the equipment level. It means we divided a petro-chemical complex into plants, next into sub-plants, sub-plants into processes, process into pieces of equipment. The implementation will be bottom up, which means we are going to model the equipment first, then the process, then the sub-plants, and so on. In this project we have implemented until the process level because our focus at this moment is to apply the Life Cycle simulation at the process level.

The petro-chemical complex produces more than one kind of product. The complex has one of more plants; each with one or more sub-plants and in each sub-plant there is one process or more processes. Figure 6.1 describes the structure in the UML notation.

In the case of this thesis, we only have to implement to scheme from the process level to the equipment level. We are going to model the process and the equipment. The higher level will be left to the future research and pilots.

6.4.1 DATABASE MODEL FOR EQUIPMENT

We will focus more on the process and the equipment. Before we model the equipment we must first gather all information needed for the pieces of equipment. We can split the attributes for the equipment according to their sources. There are three basic sources for this information, *the steady-state simulation*, *the data acquisition program,* and *the financial program.* Let us say that in this case these sources are, ASPEN, TestPoint and SAP. There are also some data that do not come from these three sources, e.g. geometrical measurement of the equipment, and we will temporarily put this into *Other Database* column.

First, we model the common equipment and later inherit this object into more specific equipment. The properties of this common equipment are also included in those pieces equipment. The process we are going to model is the *hybrid distillation/ vapor permeation* process. Not all equipment is implemented in this database model: it is confined to: *reboiler*, *distillation column, condenser*, and *membrane unit.*
The following table displays the list of attributes that are important for the equipment.

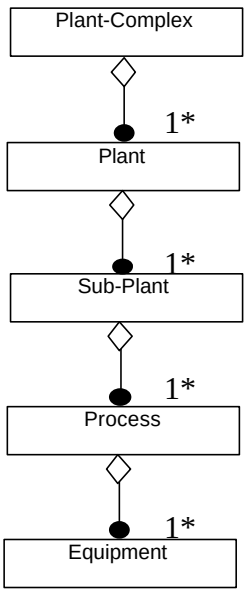


Figure 6.1 UML scheme for global process plant

Table 6.2 The needed information for a common equipment and their sources.

Common Equipment	ASPEN	TestPoint	SAP	Other database
Ident_Number	x	X	X	
Plant_Number			X	
Plant_Partition			X	
Location			X	
Active_State				X
Degradation				X
Corrosion		X	X	
Contamination		X	X	
Life_Time_Expectancy			X	
Cleaning_Cost			X	
Cleaning_Period			X	
Cleaning_DateTime			X	
Cleaning_Technique			X	
Cleaning_Solvent				X
Replacement_Cost				X
Replacement_Technique				X
Replacement_Period				X
Replacement_Materials				X
Purchase_Department			X	
Purchase_ContactPerson			X	
Purchase_DateTime			X	
Purchase_Invoice			X	
Purchase_Vendor			X	

Assuming that the following pieces of equipment are inherited from the common equipment and have those listed attributes (properties). Next, we display the additional properties for each piece of equipment. Table 6.3 shows the additional attributes for a reboiler. A complete list of other pieces equipment has to be found in the appendix.

Table 6.3 Additional attributes for reboilers

Reboiler	**ASPEN**	**TestPoint**	**SAP**	**Other database**
Temp_In	x	x		
Temp_Out	x	x		
Heat_Source	x		X	
Duty	x			
Power_Required	x			
Max_Temperature				X
Max_Pressure				X
Efficiency_rates				X
Steam_Temp		X		X
Steam_Pressure				X

6.4.2 UML SCHEME FOR EQUIPMENT

The next step to model the equipment is to define and scheme out the database structure. To assist this step, we have chosen UML (Unified Modeling Language) to define our database for equipment. UML itself is a tool to make an object-oriented model. A complete documentation about UML notation can be found in *http://www.rational.com/uml*.

There are a few notations for object modeling besides UML like OMT, Booch and Objectory. UML itself is made based on these three modeling notations. The reason why we choose UML over other three notation is mostly for the **simplicity**. Compared to the other three notations UML is simpler. Few elements in other notations have collapsed into one, for example the *active/ persistent objects* in Objectory has been replaced by *stereotype* concept in UML. The *data flow* in OMT is replaced with use cases in UML.

We will first define the object *equipment*. We can split the information about *cleaning, purchasing,* and *replacement* apart as different objects because these data will come from other sources and in practice the data may come from different departments of a production company.

Figure 6.4 shows the UML scheme for the database. The objects within the dotted box are objects that are implemented in this project. The rest of the objects are to be added in the future. The Complex would be the whole set of plants And each plant produces at least one major product. In each plant there are process sequences in which each sequence is done in a sub-plant. In these sub-plants we will see that there are one of more processes.

In every process there is at least one piece of equipment. In this context we are only

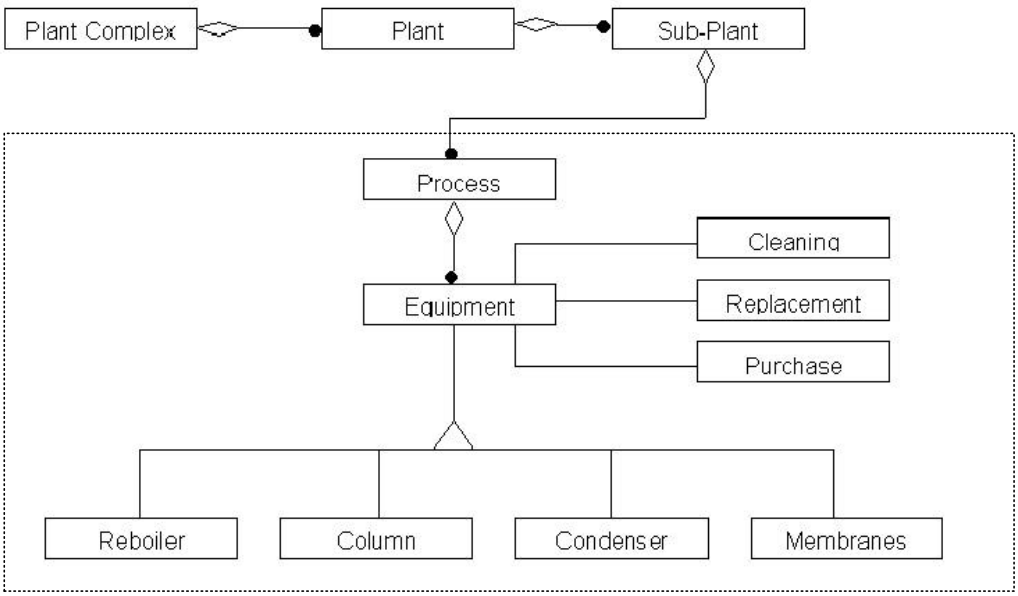


Figure 6.4 The UML scheme of the process database

about to model those pieces of equipment used in the pilot project hybrid distillation/ vapor permeation process. In our model the object ‘equipment’ is specialized in four kinds of equipment which each has its own unique properties.

This diagram is not complete with all attributes. In the complete UML diagram the attributes (actually the term *properties* is more properly used here, since an *object* has *properties* rather that *attributes*) and methods are included. Each block in the diagram can be expanded with its list of properties and the types and the methods used in the object.

The methods that are added to the model support programming and save the information into the file system. Because implementing these object databases in Delphi requires quite basic programming implies that Delphi does not provide standard functions to save object instances. The programmer has to provide ones himself.

The UML diagram brings a conceptual model for the database closer to the implementation. It is just a helpful step from a sketch to the program code. Figure 6.4 shows us a complete scheme for one object. In this example it is the basic properties for the object class equipment.

The next job after finishing this model is to implement the model in the program code. The interface part of the Delphi unit that implements this model is also included in the appendix.

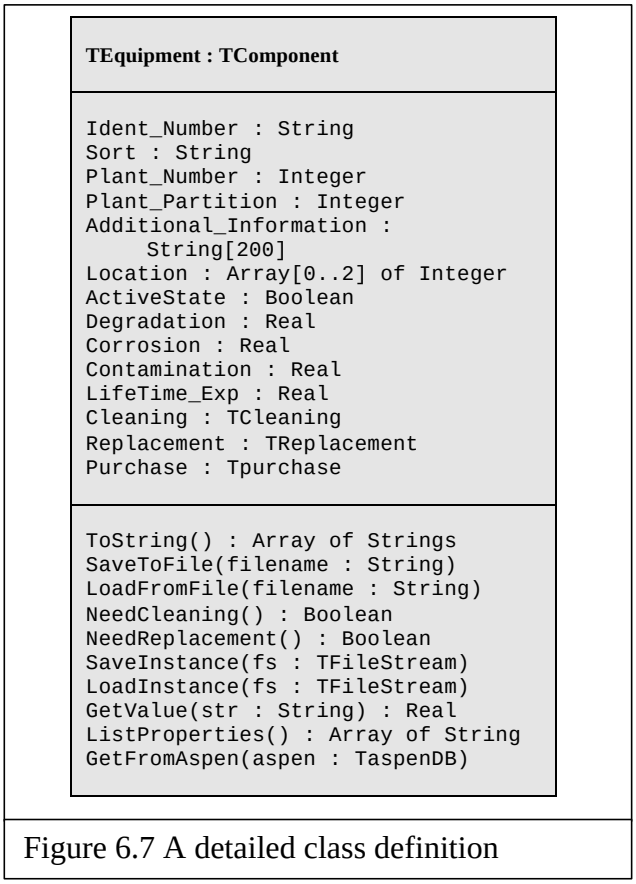


Figure 6.7 A detailed class definition

6.5 CONCLUDING REMARKS

The database module has an important role for this simulation since the simulation uses a lot of *varieties* of data sources, with different kinds of formats and as well in large quantities. In order to maintain the *integrity* and *uniformity* of the information this database module is built into the simulation. It has as well the objective to support *standardization of* the information.

For this simulation we have constructed an object-oriented database. An object-oriented data structure and database program offers more flexibility compared to the relational databases. The object-oriented database with its *inheritance* concept provides more degrees of freedom to model additional object classes in the future. Another advantage of the object-oriented database is the inclusion of *methods* of the object class, the built-in data handling procedures within the database itself This feature adds to the flexibility of the Simulation system and will speed up data manipulation.

7 DEVELOPING THE BAYESNET SIMULATION INTERFACE

7.1 INTRODUCTION

The Bayesnet Simulation application that is developed is running under Windows 98. It was tested on AMD-200MHz processor and 32MB memory. On this processor, this application is running a little bit slow, especially when processing calculations. A faster processor and a bigger memory are recommended when testing this application.

The user interface is graphical and was designed to be user-friendly. The resolution of the monitor used to develop this application is 1024x768 pixels. The user-interface was as well designed to be used in that resolution. A higher resolution will be better, a lower resolution is not recommended.

7.2 THE DEVELOPMENT OF THE INTERFACE

Before this application can run under the operating system, an external module has to be installed. That is SmileX, a module that can be downloaded from the web site of DSL. In order to use this application we assume that the relevant database and the Bayesian network model file are already saved on the hard disk. The database file is momentarily built with a script in Delphi. A database builder with graphical user-interface for this database will be planned in the future.

7.2.1 THE BASIS FOR DEVELOPING THE INTERFACE

When developing the system, emphasis was put on how the data and the results from the system would be displayed. Several aspects that we looked at, were:

- Make the system so transparent as possible to the user
- Assume that the user would be a technically oriented manager
- Assume that the results should be easy to interpret

- Assume that, when results would be depicted, the user would like to check details to confirm the assessment
- Assume that the user should be able to trace the results of all the steps:
  - The first modeling steps
  - The subsequent settings of the uncertainty
  - The actual data that are loaded
  - The details of the actual key data
  - The layout of these data should be similar to the results of functional applications

Most of these assumptions could be integrated in the design. However, in some aspects, the assumptions could not be met.
Besides, later on during the development, additional aspects were discussed and entered into the system. These were:

- Add a 3D model to illustrate the system that is simulated
- Add the ability to set the link to applications to enable users to expand the data set
- Add the possibility to graphically provide clear advise to the user

How these different aspects were integrated in the design, will be detailed in the rest of this chapter.

7.2.2 THE OVERALL SET-UP OF THE INTERFACE

Related to the assumption to provide a transparent interface, the application was designed to show all aspects of the application instantly:

1. Show the different variables that would constitute the Bayesian model.
2. Show the data that would be used.
3. Show the ability to check details.
4. Show the uncertainties that were set.
5. Show the results.

Therefore, a main window was designed, that illustrates several parts as well as the selections that are available. The selections of the databases and models are the first selection, guided by a pull-down menu like most current applications.

The main windows are divided in 3 upper sections and two lower sections. Each section points at the functions, Bayesnet performs. Upper-left, one sees the nods of the Bayesian network. The principle to call for details is included. The nodes can be clicked upon to call up details.

In the middle there is a graph showing the distribution graph of the data field.
Since it is directly linked to the nods, it enables users to check whether the data behind the nods are understandable to the user and whether the assumptions that are made are in line with the experience of the user. For example, is the “flow-rate”, that is a variable in the Bayesian network, in line with what can be expected in value as well as the spread of the value, related to the different times, the value was metered.

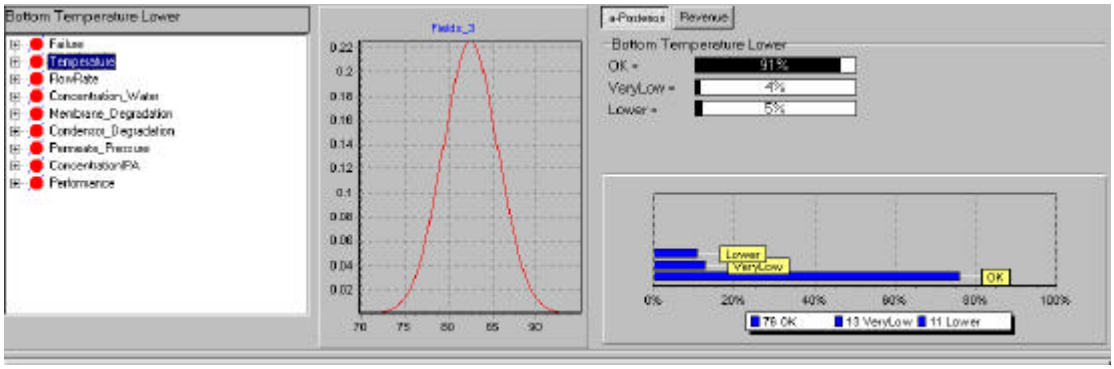


Figure 7.1 Main window of the simulation application

That being a normal distribution, the user as well can interpret the measured values that are then going to be used to make the assessment.

Thus, this graph will appear only when the selected node in the node list is associated with a data field. This is illustrated in figure 7.1. For example, the graph in figure 7.1 shows the distribution of the *top temperature* data. The *top temperature* varies around the 83°C. When the selected node in the node list has no association with a data field then the graph will be empty.

And on the most right upper side, there is a section, which shows the prior distribution of each node and its posterior distribution. This as well enables the user to directly understand the assumed prior outcomes and the corrected outcome, after the Bayes Simulation is performed. This will provide a clear idea about the "learning and correcting" outcome of the calculation.
The black bars display the prior distribution of the selected node, and the graph below it shows the posterior distribution. The posterior distribution will only be shown when the data have been analyzed.

Just above these bars we can choose between showing the distribution bars or the *Revenue* bars (see figure 7.2). The *Revenue* bars show what is the prior revenue, before the analysis, and what the assessed revenue is, after the analysis is executed and what the expected revenue in the future will be. Placing this bar to the right is in line with the usual movement of the user.

The last part of the main window is the data table. This section is actually divided into several tabs. Figure 7.3 shows this section more clearly. The tabs can be found at the bottom side of the table. These tabs are:

- Data Sheet
  This sheet displays the incoming data from the data acquisition computer. In this tab just above the table, there is a combo box that shows the selection of data sets. We can choose which data set we want to analyze.
- ASPEN Sheet
  This sheet displays the simulation data from ASPEN.
- SAP Sheet

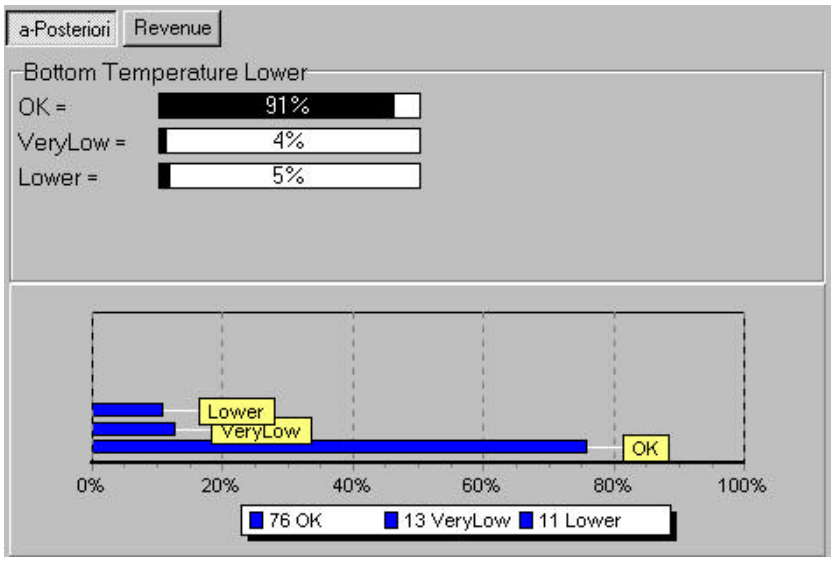


Figure 7.2 The node states graph of the main window

This sheet displays the information that was gathered from SAP. This information is momentarily a dummy, since the real data from SAP are not yet available.

- Analysis
  This sheet displays the analysis result.

7.2.2 THE DATA DISPLAY: VISUALIZING INCOMING DATA

There are basically three kinds of data needed for this application. These three databases are shown in the main window as grids. Tabs are made to jump from one database to another. The purpose of these grids is to keep the clearness of the data; the user can inspect the measurement data, or the reference data anytime. The three databases shown in this grid are:

- Process Database File.
  This database file includes the process information, the equipment data (included its simulation data) from ASPEN and association with the data field. Association to the data field means to which field a variable of the process is associated, for example top temperature associates with the 3rd field of the data record.

- Process Measurement Data File.
  This file contains information about which data files that are used for this simulation are already saved in the process database. The process data themselves are separately saved from the process database. The process database only provides the association. Thus, the actual data can be saved in several files.

- Financial and Maintenance Data File.
  At this moment the simulation has not yet supported with this file, which in the future will be provided from SAP system.

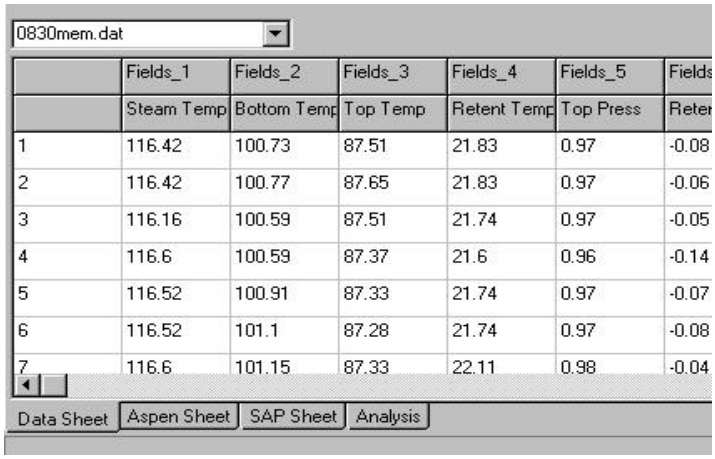


Figure 7.3 Data tabs of the main window

The tabs at the bottom of the grids can be used to choose between pages. Besides the three databases, the result of the analysis is also put in one of these four pages. This is the one most to the right. The analysis result of the simulation will only be shown as text in a grid (no visualization in this page). The visualization of the result of analysis is made in the first section in the upper half of the main window, to the right. In that part, a graph shows the results in a Revenue graph.

7.2.3 ADJUSTING THE PARAMETERS

In the design of the application, flexibility towards incoming data was envisioned. The interface had to enable a selection of data and how they would be used as parameters in the calculation and visualization. Therefore, the interface has been set-up to enable the user to adjust and expand the simulation.
When simulating a process for the first time, the user has to load a GeNIe file (*.dsl). This file has no information about which nodes correspond with which process variables. This application has a facility to make the association of the nodes to the process variable. This window has the functionality to connect the nodes of the Bayesian model to the measurement data. Connecting the nodes to the measurement data means to translate the measurement data into Bayesian terms, such as **Temperature** is **Lower**.

It is set up such that the user must first load the process database into the simulation (File | Open Database), then the GeNIe file (File | Open Model). A dialog box will appear and ask for a *.bay file by default. We can change the file type by selecting **Network File** in the **Files of Types** text box.

When both the process database and the network file are loaded we can adjust the parameters by opening the *parameters* window (Edit | Parameters).

The *parameters* window is divided into two sections, the node list and the node properties on its right side (see figure 7.4). The node list shows all nodes available in the network file.

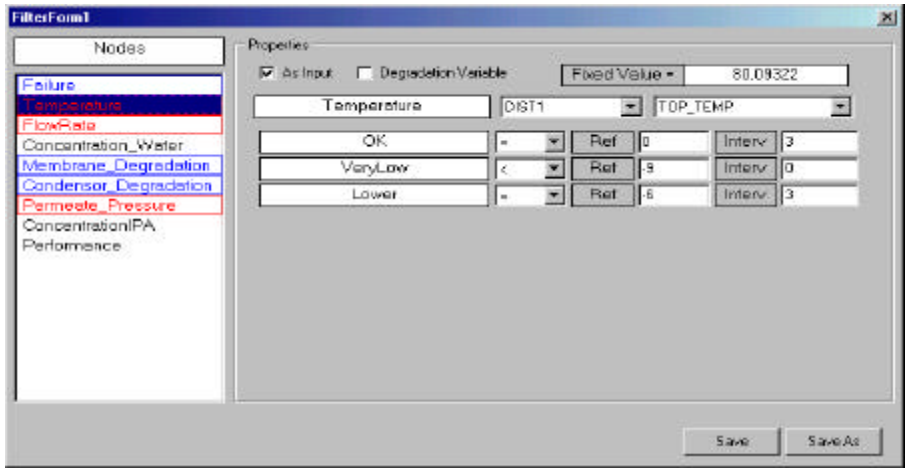


Figure 7.4 The *Parameters* window

There are three kinds of nodes in the list. These are:

- Input Node
  The input node is colored red. To make a node as an input node, check the check box *As Input* in the properties section. When this check box is checked the two combo boxes *must* be adjusted by selecting the equipment name and the process variable. As these equipment and variable are being adjusted the *Fixed Value* will change too. The next step is to define the node states. These node states depend on the *fixed value*. For the syntax for these properties, please see section 5.3.2.

- Degradation Node
  This degradation node is colored blue. To make a node as a *degradation* node, check the check box *Degradation Variable*. Furthermore, it is necessary to select a piece of equipment that corresponds with the node and the process variable *must be* filled ***Degradation*.**

- Normal Node

When a node is neither an input node nor a degradation node, it must be a normal node. It has a normal color (black).
The coloring of the data fields/windows was later added to enhance an understanding of the dataset that was used, thus enabling the user to check the process.

This approach should also enable users to store and later check and recalculate the simulation. For that matter, after filling the properties of the nodes, we can save this setting into a file. We can choose **Save As** if we want to save it into a different file (from the previous file name), then the program will ask the user for another file name.

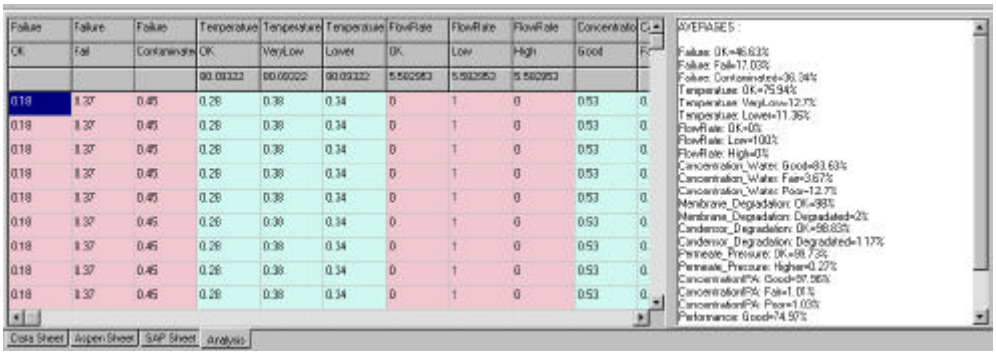


Figure 7.5 The *Analysis Result* tab of the main window

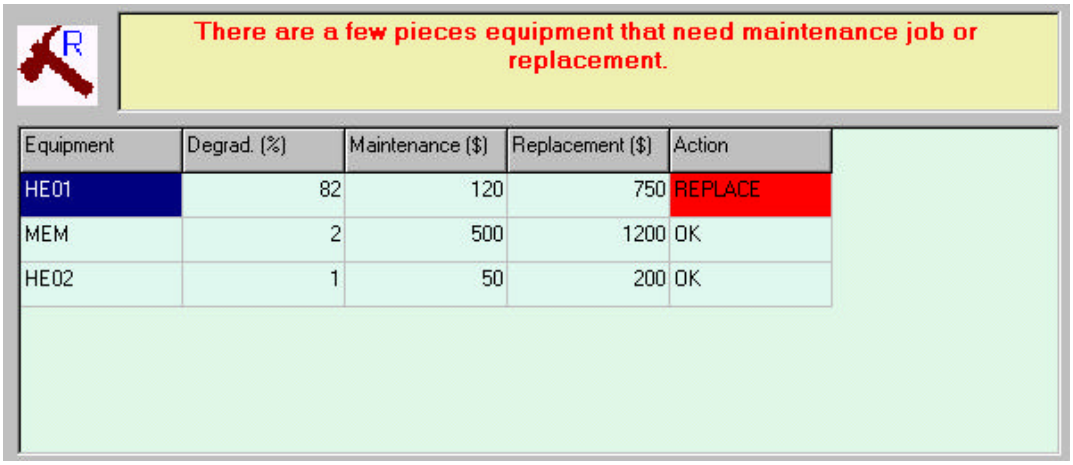


Figure 7.6 The action box and the action table.

7.2.4 ANALYZING DATA

If all data are available, we can now run the analysis. We assumed that the outcome should be simple in terms of alerting management and more detailed in terms of allowing a check of the input and output. The result of this analysis is shown in the **Analysis** tab, at the bottom of the main window.
The column of the table will spread widely and shows all states of all nodes available. Each node will be grouped with a *red* and *green* background color. The first row contains the names of the nodes. The second row contains the state names. When a node is used as input node, then its fixed value will be shown in the third row. The values of the states are then shown in the rest of the rows.

Since the overall depiction of the effect of the degradation on Revenue was not informative enough, we added a graph that shows the degradation of all equipment items in the process, by depicting a graph with information of the degradation on each piece of equipment as is illustrated. This is illustrated in figure 7.7.
When the data are analyzed, the result is averaged and shown in the text box next to the table.

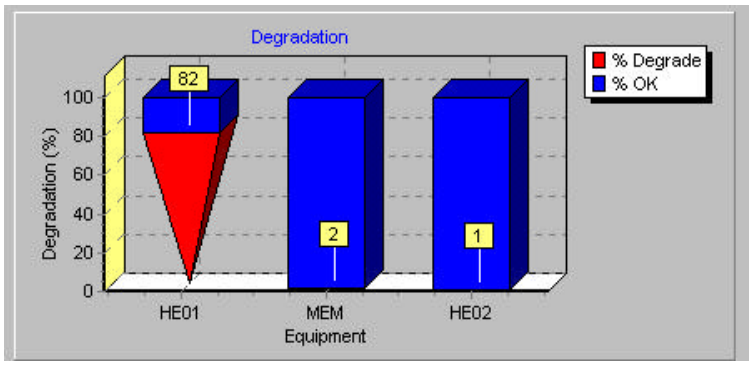


Figure 7.7 The degradation graph and performance meter

Providing a graphical feed-back in terms of a revenue graph and the depiction per equipment item did help in alerting but did not provide a detailed advise to different professionals, that would have to act on this information. Therefore, later on, an advice interface was added.

A visualization of these results is made in the advising window (see next section).

7.2.5 ADVISING WINDOW

As mentioned, an advice window was added. Assuming that several users would be especially interested (and only interested) in this result, an independent panel was designed, that could float over the other windows.
Therefore, as soon as the data have been analyzed, the *Advising* window will appear on top of the existing panels. This window visualizes the result of the analysis and gives a summary of the results. This window can be shown anytime after the data-analysis by clicking the **Advice** button just bellow the pull-down menu.
There are several sections in this window.

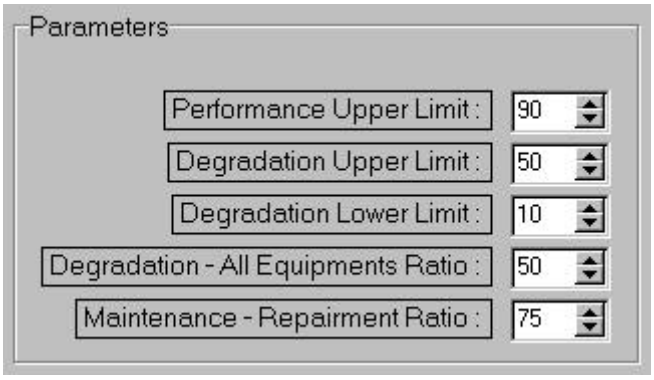


Figure 7.8 The parameter section of the advising window

The degradation graph (figure 7.7) visualizes the degradation of each piece of equipment that is listed in the process database. For each piece, there is a stacked bar graph showing the percentage of **OK** state versus the **Degradation** state of the equipment.
Next to the degradation graph the overall degradation is calculated. This is the average degradation of all equipment. The more left is the pointer is the better for the equipment. The performance meter is based on the ***Performance*** node in the Bayesian network. The more right is the pointer the better the performance is.

The **Action Box** and **Table** show the action that should, based on the degradation, be taken. This table shows as well the estimated costs for the maintenance and for the replacement for this equipment.

Assuming that the user wants to check out different aspects of the simulation, the user needs to be able to go right to the parameter setting, which thus needs to be a part of the user-interface instead of being a part of a settings file. Therefore, the last section of the advising windows is the **Parameter** section. In this section the user can change the parameters.

7.3 CONCLUDING REMARKS

To conclude this section, the first set-up of panels was experienced as clear and helpful. Later on, more detail was provided to the user to allow for more detailed information and enable a change of parameter setting.
Later, the Advise window was added on request. We had not enough underscored the requirement to provide feedback to operational users who, different from managers, wanted a clear advise on what to do. That has shaped the interface in form and in user-friendliness.

8 RESULT PRACTICAL EXPERIENCE OF SIMULATION AND DISCUSSION

8.1 TESTING THE SIMULATION

The simulation was tested based on a calculation made in ASPEN. This calculation has been made for the pilot project *Hybrid Distillation/ Vapor Permeation*, a combination of distillation process and selecting vapor components by letting the vapor through the membranes [Fakhri, 1999]. Calculation has been made by Fahkri at the API (Apparatenbouw en Processindustrie/ Laboratory of Process Equipment) at Delft University of Technology. The focus of this experiment was on finding the behavior of the process in the membranes. The goal is to break the azeotropic limitation by combining the result of distillation process with membrane technologies.

The pilot plant was built on a lab scale and for experimental purposes. There is no continuous production in this pilot. Therefore it has limitations:

- The measurement instruments are used to examine the process conditions.
- The process was manually controlled, there is no control system attached to the equipment.
- There is no continuous quality control of the product. A sample of the product was taken to the lab to determine the quality.
- Because this pilot plant was built to examine the behavior of a process, it is made under the circumstances that all equipment is properly working and no actual degradation of the equipment took place.

Due to these limitations there are several assumptions that have to be taken concerning the result of this experiment:

- All pieces of equipment are in good condition

- The quality of the product is as the experiment's report noted.
- This simulation is based on processes without control systems, thus process conditions are not automatically adjusted until manual intervention.
- This simulation is applied under stable state of the process. Data on the start-up and the shutdown may result in improper advice.

The next section will discuss the result of the simulation tests that is based on limited data, partly coming from start-up conditions.

8.2 SIMULATION RESULT

The simulation was tested with eight sets of data. These measurements were taken at several times in a day. The simulation result is shown in table 8.1.

The data files are named:

0830mem.dat contains 5728 records
1130mem.dat contains 878 records
1210mem.dat contains 4750 records
1445mem.dat contains 987 records
1530mem.dat contains 2598 records
1545mem.dat contains 1886 records
1630mem.dat contains 23814 records
1700mem.dat contains 10188 records
1830mem.dat contains 8 records

Table 8.1 shows the averages of the results for each node from every data file. The first column lists the nodes of the network. For each node the states are listed in the column next to it (e.g. for Temperature → OK, Very Low, and Low). The next column shows the prior prediction of the each state in percent. The next columns show us the results of the simulation, the posterior knowledge required from the Bayesian model. These numbers are also presented in percent. Some of these nodes are use to measure the degradation of some pieces of equipment, the Reboiler Degradation, Membranes Degradation and Condenser Degradation.

We are interested in the results of simulation about the degradation of the equipment. Figure 8.2 shows us the graph of the degradation of the pieces of equipment in each measured process data. The performance of the three pieces equipment is shown in three bars. These bars are grouped to which data file they come from. The left bar is the performance of the reboiler, in the middle the membranes and on the right is the performance of the condenser. As shown in the graph, the performances of the three pieces of equipment are almost all stable, heading to the 100% performance.

According the assumptions in the previous section there should not be any degradation. But as we see the result from the data file **1630mem.dat** we see that the degradation of reboiler reached 6.10% and probably 27.2% contaminated. For the rest

the contamination probabilities are quite low and nothing is wrong. This data set contradicts the assumption; there must be an explanation for this.

First let us take a look at the distributions of the input nodes in the file **1630.dat**. The input nodes in this simulation are: *Top Temperature, Distillate Flow Rate* and *Permeate Pressure*. From table 8.1, we can see that the distributions for these nodes in the data sets are:

Top Temperature: OK – 99.80%
Flow Rate: OK – 12.10%
Permeate Pressure: OK – 99.30%

The distillate-flow-rate in that process shows a very low percentage. Eighty four percent of the data measured in that data file shows that the flow rate is below the expected distillate-flow-rate made by ASPEN. Table 8.3 shows the comparison of the input variables and their variances. The most left column shows the measurement data that are connected to the Bayesian model. Two data are presented from the measurement data file; they are the averages and its variance. The gray shaded column shows the value that is taken from process simulation program, ASPEN. The next columns are the averages and the variance from each data file. This table is made from three measurement data files and repeating it self twice for the other six measurement data file.

Analyzing data from data file 1630 from table 8.1, we can conclude that:

1. According to the simulation, 84% of the flow rate measurements are below the expected value in ASPEN. ASPEN gave a norm value of 5.58 KG/Hr while the average flow rate of in this process was 1.44 KG/Hr.

2a. The top temperature of this process is good. The simulation calculated that 99.70% of the measurement gives OK for the temperature. It means that there was no problem with the temperature. The temperature of the column has a direct relation of reboiler. If something is wrong with the reboiler then the temperature will be influenced. The simulation concludes that the reboiler is contaminated instead of failing. Because when the reboiler fails then the temperature will drop and when it is contaminated then the top temperature will become slightly higher.

2b. From the amount of records in this file we see a very large amount of data, 23814 records. There is a possibility that the process was not running continuously while the data acquisition was online the whole time. This may also cause the distillation process not running continuously so that the average is low. This theory can be confirmed by looking at the high variance of that data. The average flow-rate was 1.44 Kg/hr, while the variance was 4.147 (see table 9.3).

Outcomes 2a and 2b are two alternatives. It is more likely that the choice goes to the later one.

		prior	0830mem.dat	1130mem.dat	1210mem.dat	1445mem.dat	1530mem.dat	1545mem.dat	1630mem.dat	1700mem.dat	1830mem.dat
Reb. Degr.	OK	95.00	98.40	99.60	98.70	99.30	99.80	99.20	66.70	99.90	100.00
	Fail	2.00	0.30	0.10	0.30	0.10	0.00	0.10	6.10	0.00	0.00
	Contamintated	3.00	1.30	0.34	1.00	0.60	0.20	0.70	27.20	0.10	0.00
Temperature	OK	90.10	99.60	99.70	99.80	100.00	99.50	99.80	99.70	99.80	100.00
	Very Low	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00
	Low	0.50	0.40	0.40	0.20	0.00	0.50	0.20	0.10	0.20	0.00
Flow Rate	OK	93.70	84.40	99.10	96.90	98.30	99.70	98.10	12.10	99.90	100.00
	Low	5.30	4.00	0.90	3.10	1.70	0.30	1.90	84.90	0.10	0.00
	High	0.00	11.60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Water Conct	Good	93.90	98.90	98.90	98.90	99.00	98.90	98.94	98.80	98.90	99.00
	Fair	1.90	1.10	1.10	1.00	1.00	1.10	1.00	1.00	1.00	1.00
	Poor	4.20	0.00	0.00	0.10	0.00	0.00	0.00	0.20	0.00	0.00
Mem. Degr	OK	98.00	98.00	98.00	98.00	98.00	98.00	98.00	98.00	98.00	98.00
	Degraded	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
Condenser Deg	OK	95.00	98.60	98.80	98.70	98.90	98.70	98.70	98.50	98.80	99.00
	Degraded	5.00	1.40	1.20	1.30	1.10	1.30	1.30	1.50	1.20	1.00
Permeate Press	OK	94.00	99.40	99.60	99.50	99.80	99.60	99.60	99.30	99.69	100.00
	Higher	6.00	0.60	0.40	0.50	0.20	0.40	0.40	0.70	0.30	0.00
IPA Concent	Good	91.30	86.00	92.90	93.00	93.00	92.90	93.00	97.20	92.90	93.00
	Fair	7.10	8.30	5.10	5.00	5.00	5.10	5.00	1.60	5.10	5.00
	Poor	1.60	4.70	1.00	1.00	1.00	1.00	1.00	1.10	1.00	1.00

Table 8.1 Simulation result from nine data sets

		ASPEN	0830mem.dat	1130mem.dat	1210mem.dat
Top Temperature (Celsius)	Average	80.09	80.50	80.50	80.50
	Variance		0.29	0.26	0.26
Permeate Pressure (Bar)	Average	1.00	1.02	1.02	1.02
	Variance		0.00	0.00	0.00
Distillation Flow Rate (KG/H)	Average	5.58	4.02	3.00	2.48
	Variance		85.06	0.10	0.08

		ASPEN	1445mem.dat	1530mem.dat	1545mem.dat
Top Temperature (Celsius)	Average	80.09	80.48	80.19	80.51
	Variance		0.19	0.34	0.00
Permeate Pressure (Bar)	Average	1.00	1.02	1.00	1.01
	Variance		0.00	0.00	0.00
Distillation Flow Rate (KG/H)	Average	5.58	2.53	3.00	3.06
	Variance		0.07	0.04	0.40

		ASPEN	1630mem.dat	1700mem.dat	1830mem.dat
Top Temperature (Celsius)	Average	80.09	80.94	80.30	80.50
	Variance		0.05	0.00	0.00
Permeate Pressure (Bar)	Average	1.00	1.02	1.00	1.00
	Variance		0.00	0.00	0.00
Distillation Flow Rate (KG/H)	Average	5.58	1.44	1.62	3.78
	Variance		4.15	5.25	3.18

Table 8.3 The averages and variance of input variables

8.3 CONCLUDING REMARKS

Analyzing the nine data files, we have assumed that these were data from stable experimental processes. We have also assumed that the pieces of equipment used in this process were in good condition. In the simulation we got the result as assumed (expected) besides for one data file. In the previous section we discussed about the possibility of the unexpected results.

Because of this pilot plant was experimental and no continuous production was made there are no actual data available for this simulation when the installation was in

actual use. Another limitation of this project was the measurement that was actually meant for experimental purposes that slightly differ from the commercial purposes. Some important quality measurements were was not included in this project, like the on-line measurement of the concentration. Measurements like these are very useful for the accuracy of the simulation calculations.

Focus on this project was to initiate a simulation for Life Cycle cost on a laboratory scale. And this application has shows that the simulation has worked properly. How it can improve the performance will be discussed in the next section.

8.4 FURTHER RESEARCH

As stated above, this application is an initial project for the Life Cycle cost simulation. This application will be applied on larger processes in the future. Further research of this Life Cycle simulation will be:

- How can the simulation become more accurate for a larger process?
- The modeling process for the Bayesian network was based on learning the process backed up with lots of substantial expert review. In the future, the modeling can be more statistically supported by preliminary correlation analysis. An expert review will still be needed for acceptance and fine-tuning the model.
- Building a simulation that is as well based on on-line process data.
- To make it is more and more compatible with commercial software packages and provides more flexibility for the user to use other software packages.

9 SHORT RESUME AND FINAL REMARKS

9.1 THE LIFE CYCLE DECISION AND ITS VALIDITY

Based on experiences in Engineering-Management of Process plants Seaview started to develop a simulation approach for the Life Cycle decisions with regards to maintenance and revamps. This inclination was based on questions from O/O's and ECP's with regards to how to manage Life Cycle Engineering and maintenance decisions.

Every process and related installation has a life-cycle that is based on a process design. The Bayesnet Simulation covers that. Since ASPEN is commonly used for chemical process industries, for the sake of argument we assume that the prior assessment for the performance and yields are based on ASPEN. Next, the process is engineered, built and commissioned. The first operational results will come in. Thus, a prior expectation can be checked with some results with regards to the performance of the process. Similarly, we assume process data and SAP data.

As is illustrated, these prior data and actual data are then fed into the simulation calculation and the expectations with regards to yields or performance are corrected.
The workings of this process of re-assing prior data with new outcomes is illustrated as well. Sometimes, the degradation will show much more prevalent than expected thus other maintenance procedures and revamps have to be discussed. The depiction of the impact in yield will help managers in setting priorities.

Thus, it has been illustrated that the data can be sufficiently retrieved and analyzed and that the Simulation can be applied in real-time situations.

9.2 THE SCOPE OF THE SIMULATION PROJECT

This project has to be seen as a pilot phase of the whole Life Cycle simulation project. The simulation was applied on a experimental process pilot plant (API) at Delft University of Technology. The process is a hybrid distillation and vapor permeation process to separate a mixture of water and isopropanol (IPA).

The simulation used the available measurement data of the experimental process and the reference data from a process simulation package. The kernel of this simulation is a Bayesian network. This network technique is based on statistical calculations and its reasoning structure can be easily recognize and traced. It implies that an IT person that has a little knowledge about a process that has built a model on a process can easily submit his model to an expert. This expert can intuitively confirm or check for a mistake in the model. This process seems to be adequate.

The Bayesian network is one of the techniques representing knowledge with uncertainties. The *fuzzy logic* in contrast, has *philosophically* mistaken the term uncertainty by literally *blurred* (make fuzzy) the logic variables. The Bayesian technique holds the logic rules and gives the better representation of the joint probabilities.

During this pilot, a great deal of time was spent to built a database for the process. This simulation has many different data sources. Besides the process and measurement data, it will cover *financial*- and *maintenance data*. To integrate these data, it is necessary to build a database based on the process. Integration of data will mean a) bringing all the data files together to a one bigger file as well as b) conversions between units and c) conversions between file formats. The second reason why building this database is to be able to standardize. This will happen when the format conversions between the data file takes place.

The last part of this simulation is an added rule-based advising module. This module is still in a prototype phase. At this moment, this rule-based advising module will take the results of the simulation and provide some actions to be taken or to be considered. In the future this module will be more sophisticated by taking in all information available such as the detailed maintenance cost and detailed replacement cost.

The simulation has, in general terms, shown the expected result. Some of the results were not expected under a few assumptions, but these deviations have been discussed and explained. Analyzing the data, we have concluded that the unexpected results may most probably be caused by the non-continuity of this process and lack of measurement data, due to the fact that this plant was built for and run for experimental purposes.

Based on the result of this pilot, the Life Cycle simulation project will be continued. In the future this simulation will be expanded to be more flexible andto take in information from external sources. The modeling process of the Bayesian model will be more automated and then applied to different actual cases. This will provide the proper basis to fine-tune the simulation and to make this approach more effective and to develop it to be able to use it in an actual industrial process setting.

BIBLIOGRAPHY

Borgers, J.J.J. (1997). Supporting the Change in Structure in a Decision Support System Based on Structural Equations. Master thesis, Delft University of Technology

Fakhri (1999). Hybrid Distillation/Vapour Permeation Systems for Solvent Dehydration. Ph.D. Thesis, Delft University of Technology

Graham, I. (1988). Expert Systems: *Knowledge, Uncertainty and Decision*. Chapman and Hall Computing, New York.

Jarmulak, J. (1999). Case-Based Reasoning for NDT Data Interpretation. Ph.D. Thesis Delft University of Technology.

Jensen, Finn V. (1996). An Introduction to Bayesian Networks. UCL Press, London.

Kister, H.Z. (1992). Distillation Design. McGraw-Hill, Inc, New York.

Kister, H.Z. (1990). Distillation Operation. McGraw-Hill, Inc, New York.

Kroes, P. (1996). Ideaalbeelden van Wetenschap: *Een Inleiding tot De Wetenschapfilosofie*. Boom, Amsterdam.

Miller, T et al. (1997). Using Delphi 3. Que, Indianapolis.

Montgomery, S. (1998). Building Object-Oriented Software. McGraw-Hill, Inc, New York.

Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems: *Networks of Plausible Inference*. Morgan Kaufmann Publisher, Inc, San Mateo, California.

Russell, S.J. and Norvig P. (1995). Artificial Intelligence: *A Modern Approach.* Prentice Hall International, New Jersey.

Thijssen, C.P.R.(1999). SmileX: *An AcitveX Decision-Analytic Reasoning Engine and Its Application to Evaluation of Credit Applicants*. Master thesis, Delft University of Technology.

A APENDIX BAYESNET MODULE

A1 THE MODULE INTERFACE OF BAYESNET

CODE FROM THE UNIT FILE BAYESDB.PAS .

```
unit BayesDB;

interface

uses OleCtrls, SMILEXLib_TLB, SysUtils, Classes;

type TAssociator = (asEqual, asLess, asLessEqual, asGreater,
asGreaterEqual);

     TNodeState = Class(TObject)
       private
         FValue  : Real;
         FParent : TObject;
         FInterval : Real;
         FRefference : Real;
         FAssociator : TAssociator;
         function GetAssociatorIndex : Integer;
       public
         Name    : String[30];
         constructor Create(name : String; var parent : TObject);
         procedure SetRule(Associator : TAssociator; FixedValue :
                 Real;
                           Refference : Real; Interval : Real);
         function StateStatus(AVal : Real) : Boolean;
         procedure SaveInstance(fs : TFileStream);
         procedure LoadInstance(fs : TFileStream);
         property Value : Real read FValue;
         property Associator : TAssociator read FAssociator;
         property AssociatorIndex : Integer read GetAssociatorIndex;
         property Interval : Real read FInterval;
         property Refference : real read FRefference;
     end;

     TBayesNode = Class(TObject)
        private
```

```
      FASInput : Boolean;
      FASDegradation : Boolean;
      FName    : String;
      FNet     : TSmileX;
      FStateCount : Integer;
      FStates  : Array of TNodeState;
      FVariable : String;

      function GetStateName(idx : integer) : String;
      function GetStateValue(idx : integer) : Double;
      function GetState(idx : integer) : TNodeState;
  public
      Tag1       : Integer;
      Tag2       : Integer;

      constructor Create(name : String; var Parent : TSmileX);
      destructor Destroy; override;
      function GetStatusIndex(AVal : real) : Integer;
      procedure SetVariable(AVar : String);
      procedure SaveInstance(fs : TFileStream);
      procedure LoadInstance(fs : TFileStream);
      procedure MakeEvidence(StateIdx : Integer);

      property Name : String read FName write FName;
      property ASInput : boolean read FASInput write FASInput;
      property StateCount : integer read FStateCount;
      property StateName[idx : integer] : string read
            GetStateName;
      property StateValue[idx : integer] : Double read
            GetStateValue;
      property States[idx : integer] : TNodeState read GetState;
      property Variable : String read FVariable write FVariable;
      property AsDegradation : Boolean read FAsDegradation write
                                                  FAsDegradation;
end;

TBayesNet = Class(TSmileX)
  private
      FNodes : Array of TBayesNode;
      FNodeCount : Integer;
      FFileName : WideString;

      function GetNode(idx : integer) : TBayesNode;

  public
      constructor Create(AOwner : TComponent); override;
      destructor Destroy; override;

      procedure ReadFile(const Filename : WideString);
      procedure SaveToFile(Filename : String);
      procedure LoadFromFile(Filename : String);
      procedure Clear;

      property NodeCount : integer read FNodeCount;
      property Nodes[idx : integer] : TBayesNode read GetNode;
end;
```

A2 USING TBAYESNET

1. Class TBayesNet (TSmileX)

The class TBayesNet is comparable to the object class TSmileX. This class is derived from the class TSmileX, which means that all functionality of TSmileX, are still included in the object class TBayesNet. The object class TBayesNet can be considered as a collection of nodes. These nodes are loaded from a GeNIe network file. The structure of the network can be loaded but can not be retrieved from the object class TBayesNet since TSmileX does not offer this functionality either.

Once the network file is loaded the TBayesNet object can be seen as a collection of nodes. The nodes are saved as a dynamic array. These nodes have the type of object class TBayesNode.

These are the properties and the methods:

Create (AOwner : TComponent);
Use this constructor to create an object instance. AOwner is a TComponent, and the TBayesNet instance will also be destroyed when AOwner is destroyed.

Destroy;
Use this destructor to free the memory used for TBayesNet instances. When this method is called it will also destroyed all the TBayesNode instances within the TBayesNet instance.

ReadFile(const Filename : WideString);
This method loads an (*.dsl) file into the TBayesNet instance.

SaveToFile(Filename : string);
This method saves the TBayesNet instance into a file, with its TBayesNode instances.

LoadFromFile(Filename : string);
This method loads the TBayesNet file that is made by SaveToFile method.

Clear;
This method reset the TBayesNet instance; erasing all the TBayesNode instances.

NodeCount : integer;
This property returns the number of nodes in the TBayesNet instances.

Nodes[idx : integer] : TBayesNode;
This property returns the idx^{th} node of the TBayesNet; idx is an integer from 0 to (NodeCount-1).

2. Class TBayesNode(TObject)

The object class TBayesNode represents a node of a Bayesian network. This object class is customized for the Life Cycle simulation; there are a few additional properties for the simulation's purposes, such as connecting a node as input or as output (degradation). This object class as a collection of states. Each state is an object class TNodeState.

These are the properties and the methods:

Create (name : String; var Parent : TSmileX);
This constructor is used to create a new TBayesNode object instance, but it is not recommended that the user call this constructor; the TBayesNet class will make the TBayesNode instances automatically when loading a network file.

Destroy;
This is a destructor for the object class TBayesNode. Do not call this method manually; the object class TBayesNet will call this method automatically when destroying itself.

Name : Integer;
This property returns the name of the node.

ASInput : Boolean;
This property defines whether a node is used as input i.e. can take any measurement value.

ASDegradation : Boolean;
This property defines whether a node is used as output i.e. can give probabilities of degradation.

StateCount : Integer;
This property gives the number of states in this node.

States[idx : integer] : String;
This property gives the idx^{th} state where idx is an integer from 0 to (StateCount-1).

StateName[idx : integer] : String;
This property gives name of the idx^{th} state where idx is an integer from 0 to (StateCount-1).

StateValue[idx : integer] : Double;
This property gives the value of the idx^{th} state where idx is an integer from 0 to (StateCount-1).

MakeEvidence(StateIdx : Integer);
Use this method to enter evidence into the network. StateIdx is the index of the state which value will be made 1.

3. Class TNodeState(TObject)

This class represents a state for a Bayesian node. This object is enhanced to contain a rule to convert a real value to a Bayesian value, which is in term of probabilities.

These are the properties and the methods:

Name : String;
This property holds the name of the state.

Create(name : String; var parent : TObject);
This constructor creates an object instance. Do not call this method directly; the object class TBayesNode will create it automatically when needed.
Destroy;
This is a destructor, which frees the memory. Do not call this method directly; the object class TBayesNode will destroy its states automatically.

Value : Real;
This property holds the fixed value of the state.

Reference : Real;
This property holds the reference value for the rule.

Interval : Real;
This property holds the interval value for the rule.

SetRule(Associator : TAssociator; FixedValue : Real; Reference : Real; Interval : Real);
Use this method to set a translation rule for the state. The TAssociator itself can have the values (asEqual, asLess, asLessEqual, asGreater, asGreaterEqual). The value of FixedValue, Reference and Interval will be hold by the properties Value, Reference and Interval respectively. Please see the description of StateStatus for the description of the rule.

StateStatus(AVal : Real) : Boolean;
This method returns the status of the state for a given value (AVal). The status of a state for a given value can be either true or false. It is true if the value lies on an interval that is defined by a rule. The rule is given by SetRule method. The StateStatus will returns true if any of these conditions is satisfied:

- If (Associator=asEqual) and (AVal < Value + Reference – Interval) and (AVal > Value + Reference – Interval)
- If (Associator=asLess) and (AVal < Value + Reference)
- If (Associator=asLessEqual) and (AVal <= Value + Reference)
- If (Associator=asGreater) and (AVal > Value + Reference)
- If (Associator=asGreaterEqual) and (AVal >= Value + Reference)

Associator : TAssociator;
This property holds the associator.

B APPENDIX LIST OF PROPERTIES

Each table represents object-class for the Simula database and the list of its properties. For each property will be appointed from which database it come from.

Cleaning	ASPEN	TestPoint	SAP	Other databases
Cost			x	
Period			x	
DateTime			x	
Technique			x	x
Solvent			x	x

Replacement	ASPEN	TestPoint	SAP	Other databases
Cost			x	
Period			x	
DateTime			x	
Technique			x	x
Tools			x	x
Materials			x	x

Purchase	ASPEN	TestPoint	SAP	Other databases
Department			x	
Contact_Person			x	
DateTime			x	
Vendor			x	
Invoice_Number			x	

Common Equipment	ASPEN	TestPoint	SAP	Other database
Ident_Number	x	x	x	
Plant_Number			x	
Plant_Partition			x	
Location			x	
Active_State				x
Degradation				x
Corrosion		x	x	
Contamination		x	x	
Life_Time_Expectancy			x	
Cleaning_Cost			x	
Cleaning_Period			x	
Cleaning_DateTime			x	
Cleaning_Technique			x	
Cleaning_Solvent				x
Replacement_Cost				x
Replacement_Technique				x
Replacement_Period				x
Replacement_Materials				x
Purchase_Department			x	
Purchase_ContactPerson			x	
Purchase_DateTime			x	
Purchase_Invoice			x	
Purchase_Vendor			x	

Reboiler	ASPEN	TestPoint	SAP	Other database
Temp_In	x	x		
Temp_Out	x	x		
Heat_Source	x		x	
Duty	x			
Power_Required	x			
Max_Temperature				x
Max_Pressure				x
Efficiency_rates				x
Steam_Temp		x		x
Steam_Pressure				x

Column	ASPEN	TestPoint	SAP	Other databases
Mix_Feed	x	x		
Mix_Distillate	x	x		
Mix_Bottom	x	x		
Top_Temp	x	x		
Bottom_Temp	x	x		
Press	x	x		
Reflux_Ratio	x	x		
Stripping_Ratio	x	x		
Diameter				x
Number_of_Trays				x
Height				x
Packing_Efficiency	x			x
Feed_Height	x			x
F_Factor				x

Membrane	ASPEN	TestPoint	SAP	Other databases
Feed	x	x		
Feed_Pess	x	x		
Permeate_Press	x	x		
Permeate	x	x		
Retentate_Press	x	x		
Retentate	x	x		
Recycle_Press	x	x		
Recycle	x	x		
Temp	x	x		

Material	ASPEN	TestPoint	SAP	Other databases
Ident	x			
materal_name	x			
mole_flowrate	x			
mass_flowrate	x			

Mixture	ASPEN	TestPoint	SAP	Other databases
ident	x			
mixture_name	x			
vol_flowrate	x	x		
materials	x			
mass_flowrate	x	x		
mole_flowrate	x	x		
mass_fraction	x			
mole_fraction	x			

C APPENDIX UML CLASS DEFINITION

TCleaning = class(TComponent)
Cost : Real Period : Real DateTime : TDateTime Technique : String Solvent : String
ToString() : Array of Strings SaveInstance(fs : TFileStream) LoadInstance(fs : TFileStream)

TReplacement = class(Tcomponent)
Cost : Real Period : Real DateTime : TDateTime Technique : String Tools : String Materials : String
ToString() : Array of Strings SaveInstance(fs : TFileStream) LoadInstance(fs : TfileStream)

TPurchase = class(TComponent)
Department : String Contact_Person : String DateTime : TdateTime Vendor : String Invoice_Number : String
ToString() : Array of Strings SaveInstance(fs : TfileStream) LoadInstance(fs : TfileStream)

TEquipment = class (TComponent)
Ident_Number : String Sort : String Plant_Number : Integer Plant_Partition : Integer Additional_Information : String Location : Array [0..2] of Integer ActiveState : Boolean Degradation : Real Corrosion : Real Contamination : Real Life_Time_Exp : Real Cleaning : Tcleaning Replacement : Treplacement Purchase : Tpurchase
ToString() : Array of Strings SaveToFile(filename : String) LoadFromFile(filename : String) NeedCleaning : Boolean NeedReplacement : Boolea SaveInstance(fs : TFileStream) LoadInstance(fs : TFileStream) GetValue(str : String) : Real ListProperties() : Array of Strings GetFromAspen() : TaspenDB

TReboiler = class(Tequipment)
Temp_in : Real Temp_out : Real Heat_Source : String Duty : Real Power_Required : Real Max_Temp : Real Max_Press : Real Efficiency_Rate : Real Steam_Temp : Real Steam_Press : Real Measurement_File : String miTemp_in : Integer miTemp_out : Integer miDuty : Integer miPower_Required : Integer miSteam_Temp : Integer miSteam_Press: Integer
ToString() : Array of Strings SaveInstance(var fs : TFileStream) LoadInstance(var fs : TFileStream) GetValue(str : String) : Real ListProperties() : Array of Strings

TColumn = class(TEquipment)
Mix_Feed : Tmixture Mix_Distilate : Tmixture Mix_Bottom : Tmixture Top_Temp : Real Bottom_Temp : Real Press : Real Reflux_Ratio : Real Stripping_Ratio : Real Diameter : Real Number_of_Trays : Integer Height : Real Packing_Efficiency : Real Feed_Height : Real F_Factor : Integer miTop_Temp : Integer miBottom_Temp: integer miPress : Integer miRefluxRatio : Integer Measurement_File : String
ToString() : Array of Strings SaveInstance(fs : TFileStream) LoadInstance(fs : TFileStream) GetFromAspen() : TAspenDB GetValue(str : String) : Real ListProperties() : Array of Strings

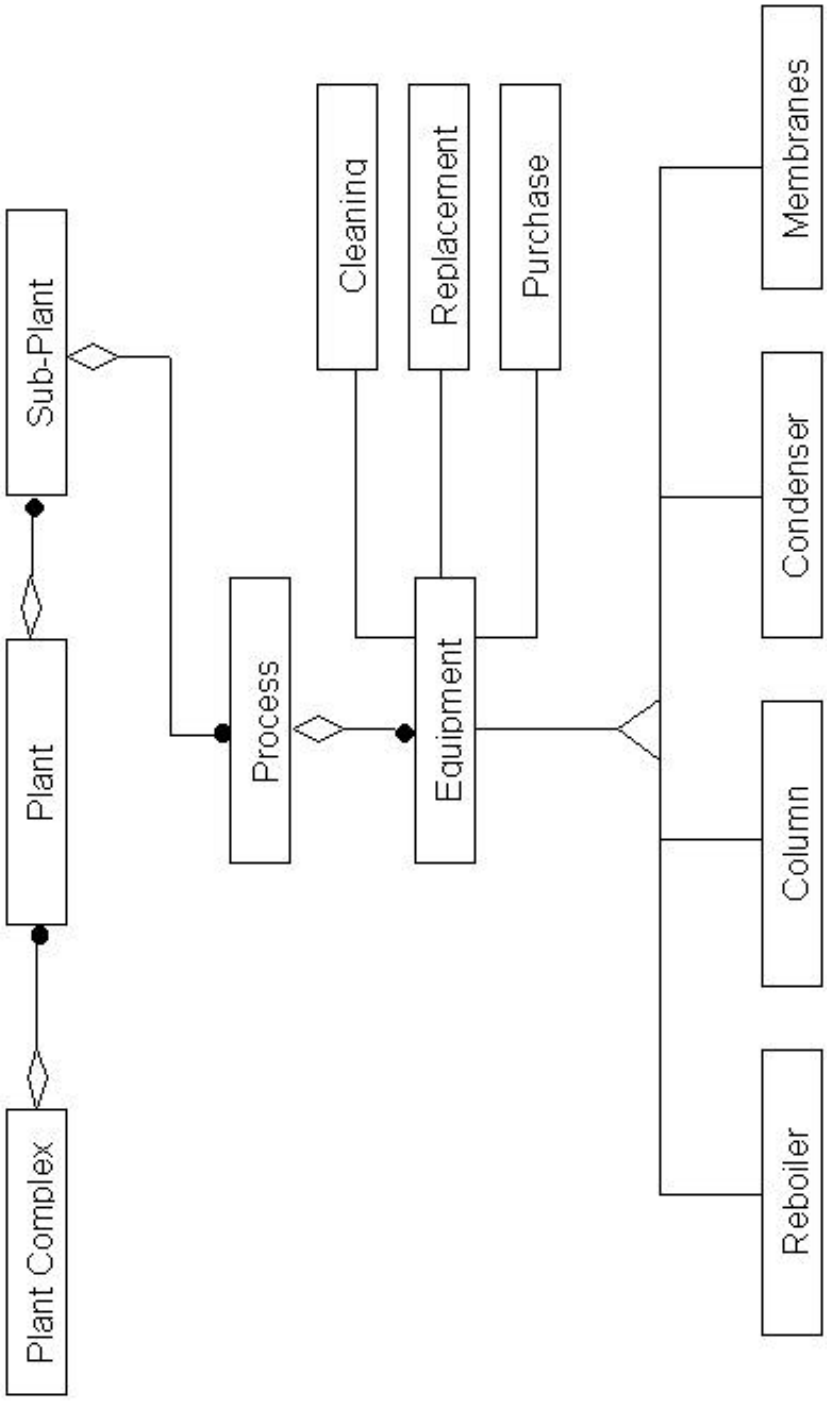
TPump = class(TEquipment)

TCondensor = class(Tequipment)

TMembrane = class(Tequipment)
Feed : Tmixture Feed_Press : Real Permeate_Press : Real Permeate : Tmixture Retentate_Press : Real Retentate : Tmixture Recycle_Press : Real Recycle : Tmixture Temp : Real miFeed_Press : integer miPermeate_Press : integer miRetentate_Press : integer miRecycle_Press : integer miTemp : integer miPress_Drop : integer Press_Drop : Real
SaveInstance(fs : TFileStream) LoadInstance(fs : TFileStream) GetFromAspen() :: TaspenDB GetValue(str : String) :Real ListProperties() : Array of Strings

TMaterial = class(TComponent)
ident : String material_name : String mole_flowrate : Real mass_flowrate : Real
ToString() : Array of String SaveInstance(fs : TFileStream) LoadInstance(fs : TFileStream)

TMixture = class(TComponent)
ident : String mixture_name : String vol_flowrate : Real miVol_flowrate : integer miMass_flowrate : integer
Material(idx : Integer) : Tmaterial mass_flowrate() : Real mole_flowrate() : Real mass_fraction(idx : integer) : Real mole_fraction(idx : integer) : Real ToString() : Array of String SaveInstance(fs : TFileStream) LoadInstance(fs : TFileStream) Clear () GetValue(str : String) : Real ListProperties() : Array of Strings


Plant Complex
Plant
Sub-Plant
Process
Equipment
Cleaning
Replacement
Purchase
Reboiler
Column
Condenser
Membranes