The use of a neural network for reliability comparison at Assembléon

Franck, P.A.L.

Award date:
2008

Eindhoven, July 2008

The use of a neural network for reliability comparison at Assembléon

by

P.A.L. Franck

Student identity number 0568773

in partial fulfillment of the requirements for the degree of

Master of Science
in Operations Management and Logistics

Supervisors:
Dr. ir. P.J.M. Sonnemans, TU/e, BPD
Dr. A.J.M.M Weijters, TU/e, IS

TUE. Department Technology Management.
Series Master Theses Operations Management and Logistics

Subject heading: Neurale netwerk, voorspellingen

Supervisor Assembléon Netherlands B.V.

Ir. P.G.H. Tackenberg
Supervisor Assembléon Netherlands B.V.
System Test and Release

De Run 1110
5503 LA Veldhoven

Supervisors Eindhoven university of Technology

First Supervisor
Dr. ir. P.J.M. Sonnemans
Department of Business Process Design

Second supervisor
Dr. A.J.M.M Weijters
Department of Information Systems

A. Balasubramanian,
PhD-student at Department of OPAC

Author

P.A.L. Franck
Student nr. 0568773
Eindhoven University of technology
Faculty: Technology management
Study: Operation management & logistics
Section: Quality and Reliability Engineering

Preface

This thesis finalizes my graduation project at Assembléon and my master Operation Management and Logistics. This master thesis focused on creating a user profile for the different Surface Mount Technology placement equipment placed all over the world. This user profile was used as input for modeling the relation between machine setup and failure rate. This project has been carried out at the sub department Quality and Reliability Engineering.

I first would like to thank my first supervisor Peter Sonnemans for his input and feedback during the project. Because this research was part of a Promotion research, I would like to thank Aravindan Balasubramanian, PHD student at Eindhoven University of Technology, for all his input and guidance during the project. I wish him success with finalizing his PHD project. Next I would like to thank my second supervisor Ton Weijters for the feedback on my report and his input, especially on the subject of neural networks.

I would also like to thank my colleagues of the department STaR at Assembléon for showing me the robes of the machines and the company but also for helping me gathering the necessary data for my project. I would especially like to thank Patrick Tackenberg for his support during my project and the useful input on user profiles.

Finally I would like to thank my family and friends for their support and all the nice times we had during my study.

Abstract

This report presents the results of the research performed at the System Test and Release department of Assembléon in Veldhoven, concerning the use of a neural network for reliability comparison. In this exploratory research a neural network is constructed, based on the data that was available at the moment. The lack of sufficient data made it impossible to construct a working neural network for prediction purposes. The research identified the different steps that have to be taken during the creation of a neural network and the pitfalls that are related to the construction of a neural network.

Executive summary

This report describes the master thesis research that is performed at the department 'System Test and Release' (STaR) of Assembléon in Veldhoven. The STaR department is responsible for the verification and validation of Assembléon products.

Assembléon develops, assembles, markets, and distributes a diverse range of Surface Mount Technology (SMT) placement equipment. With the developed and produced products, Assembléon tries to provide their customers with the most flexible and cost effective means to deal with the fast changing product mix and batch sizes that are common in the electronics industry. Customers of Assembléon use the products to place Surface Mount Devices and a large variety of electronic components on Printed Circuit boards (PCBs).

Assembléon provides a product range in pick and place robot systems which cover any application and any volume demand in the electronics assembly industry. Assembléon offers three different series of products: the A, M and X-series. In this research the focus lies with the AX-301 and AX-501. These machines are most suitable for higher placement rates between 45,000 components per hour (cph) and 150,000 cph.

The machines that are sold by Assembléon are continually upgraded with new software releases. Assembléon is searching for a method that can describe in a short period of time whether the new software version performs as reliable as or better than the previous version. With this method Assembléon can determine whether the new software version is ready for release (the reliability is equal or higher to the previous version) or if further development is needed.

This focus on determining reliability has an overlap with the research that is performed at the department of Quality and Reliability Engineering at Eindhoven University of Technology. Within this department Aravindan Balasubramanian is performing his PHD research, with the main objective of estimating and predicting reliability of the product during development and to establish a good reliability diagnostic process to help reliability improvement. To predict the reliability, an Artificial Intelligence technique called Neural Networks is used.
This master thesis is setup as an exploratory research for the PHD project of Arvindan Balasubramanian and therefore mainly focuses on the different steps that need to be taken during the development of a neural network. By performing this exploratory research, the possible pitfalls and subjects that need extra attention can be identified.

The goal of this research is to perform a feasibility study in which is investigated if a neural network can be constructed that can predict the reliability performance of an Assembléon machine, by using values of operational conditions, machine configuration, machine performance, and product characteristic. This neural network will be used to estimate the reliability of Assembléon machine running with an old software version. The reliability of new software version is determined on an actual Assembléon machine running in the field. Comparing both the reliability of an Assembléon machine running with an old and new version makes it possible to decide whether the new software version is ready for release. Figure 1 shows a graphical representation of the method to determine whether a new software version is ready for release

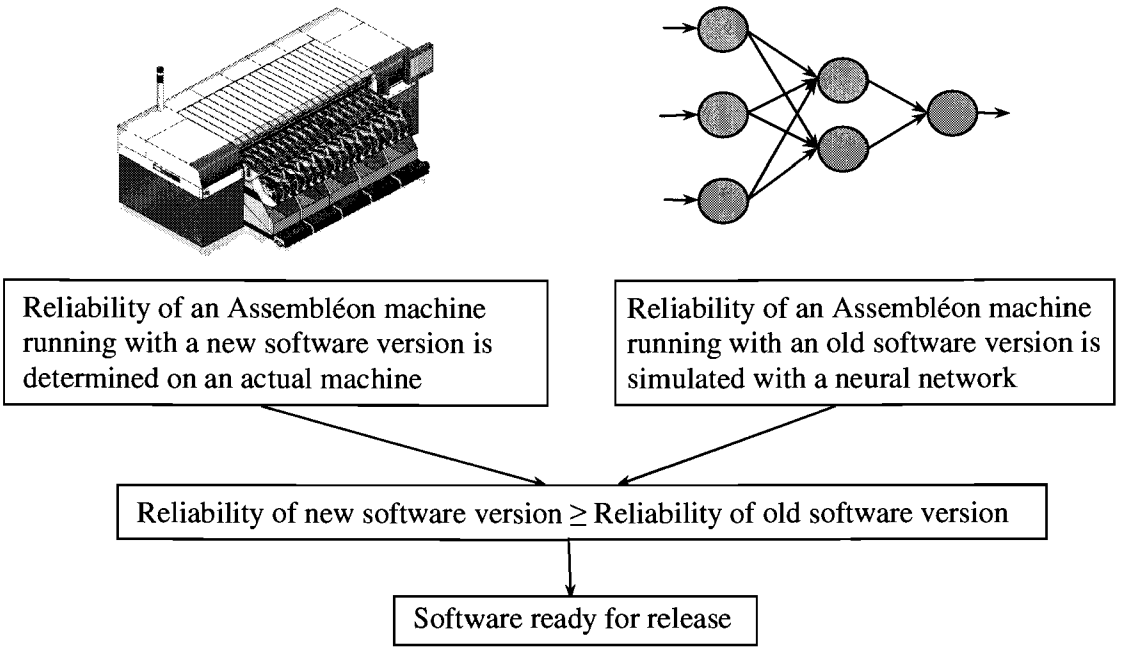


Figure 1: graphical representation of the method to determine whether a new software version is ready for release

Concerned literature

Reliability is an important attribute of a product because failure of the product can be very costly and lead to unsafe situations. Generally reliability is defined as "the ability of a product to perform a required function under stated conditions for a stated period of time" [Wal06].
There are many different tools and methods to predict reliability. A lot of these tools use complex mathematical models to predict the reliability of systems. This is a disadvantages because the mathematics that is used in the models may not be able to represent the behavior or structure of the design in the correct manner An other disadvantage of classical prediction methods, like regression analysis, is the inability to deal with uncertainty. Information that is used to predict reliability early in the development often contains a lot of uncertainty. Not incorporating this uncertainty into the reliability prediction will result in incomplete estimations of the reliability.
Neural networks are capable of incorporating uncertainty into the reliability prediction. Neural Networks (NNs) are software based constructs that use algorithms that are capable of performing functional input/output mapping. NNs are especially useful when the physical–mathematical models to be described by them, are complicated or unknown. This is because NNs acquire knowledge through learning. For this learning process a data set is necessary that contains information on the independent and dependent variables.

User profile

The collection of several indicators that represent how a product/system is used by its user is referred to as a user profile. The collection of user information in a user profile can be valuable in many stages (e.g defining which (new) features to include in a machine, setting up test programs,) of the product life cycle of Assembléon machines. In this research the user profile is used as input for a neural network that is used for predicting the reliability of an Assembléon machine.
A lot of information can be collected about the properties of a customer and how a machine is used, but not all this information is useful when predicting the reliability of the system. Therefore a brainstorm session is organized and a questionnaire constructed which enabled

Assembléon employees to express their opinion about how strong a certain variable influenced the reliability of an Assembléon machine.
The selection of which variables to include in the user profile was based on the availability of data and how practical it was to gather the information that could represent this variable. Some variables are excluded from the user profile even though Assembléon employees expressed that there could be a strong relation between variable and the reliability of an Assembléon machine.

Neural network
The development of a neural network starts with getting more insight into what the neural network is going to represent. The problem for which the neural network will be used in this research is:
Assembléon is searching for a method that can describe in a short period of time whether a new software version performs as reliable as or better than its previous version.

For the problem described above, a neural network could be the solution. But the problem definition for the neural network would be:
Predicting the reliability performance of an Assembléon machine, by using data on operational conditions, machine configuration, machine performance, and product characteristic.

The values of the different factors included in a user profile are used as the basis for the input of the neural network. The user profile contained 43 data points that could be used for training. One of the disadvantages of using neural networks is the large amount of data needed for training the network. Since only 43 data points were available it was decided to reduce the amount of factors that were included in the neural network. The reduction of factors was based on correlation between factors, the practical use of implementing factors in a neural networks and whether a factor had a constant value for all the machines in the data set. After this reduction of factors it was decided to construct two different neural networks. One would model the relation between machine configuration and the reliability of the machine and the other would model the relation between machine performance and the reliability of the machine.

To get a better idea about the predictive performance of the two neural networks, a benchmark is needed against which the predictive performance of the network can be compared. In this research a linear regression model was used to construct a benchmark. The predictive performance of the neural network is compared with the predictive performance of the regression model. When the predictive performance of the neural network is poorer than the regression model the configuration of the neural network should be altered or the conclusion should be drawn that a neural network is not an ideal tool to model the relation between input and output.

Results
The construction, training and validation of the neural network are performed in Matlab. The results of the training and validation showed large variations between training cycles. This is probably caused by the small sample size that is available at the moment. The linear regression model that is constructed as benchmark showed also a poor performance (low R^2 and F ratio).

Conclusion

Due to the large variation in the results between training cycles, the neural network could not be used for the prediction of the reliability of an Assembléon machine. Not being able to predict the reliability with a neural network also makes it impossible to use the neural network to determine whether software is ready for release. The bad performance of the neural network is probably caused by the small sample size that is available at the moment, but it could also be possible that the neural network is not capable of modeling the relation between input and output variables.

The same conclusions can be drawn from the regression analysis. The small sample size caused that the linear regression model could not identify a strong relationships between independent variables and the dependent variable (MTBF). The bad performance of the regression model could also be caused by the fact that there are no strong linear relationships between independent variables and the dependent variable, which would make a linear regression model not an ideal tool to construct a benchmark for the performance of the neural network.

Discussion

The small data set has caused an insufficient training of the neural network. The small data set also initially caused a reduction in the factors that could be included in the neural network. This reduction may have resulted in the exclusion of factors that have a large influence on the reliability of the machine, which causes a poorer predictive performance of the neural network.

The small data set is an important issue when using a neural network. Therefore the increase of the sample size is important. Several issues with the data collection make it hard to increase the sample size. Some of the issues are that the information is scattered throughout different files, the collection of this information and analysis has to be performed manually which makes the process very time consuming. The files that are needed for the data collection have to be gathered by an operator working the machine, these operators don't set priorities with the collection of this information which results in the collection of incomplete files.

The problem of time consuming data collection and analysis and the collection of in complete files by operators can be solved by developing an automatic data collection and analysis tool.

Besides automatic data collection also the possibilities should be investigated to use different prediction techniques. When information is gathered over time it could be possible to construct a time series which could be analyzed to get a better understanding of which relation exists between factors and the reliability of an Assembléon machine.

Content

1. Introduction

Our lives are filled with uncertainties. Most of us have learned to deal and live with day-to-day uncertainties and to make choices and decisions in the presence of uncertainty. In order to compensate for the effects of uncertainties we have evolved cognitive heuristics, and developed strategies. The problem of these heuristics and strategies are that they don't always perform as well as we would like them to [Mor92]. Therefore scientific methods and techniques are and have been developed to predict and deal with uncertainties.
Uncertainty plays an important role in the development of new products. Using new technologies, components, and techniques introduces a lot of uncertainties into the development process. During the development process the goal is to develop a reliable product. In order to develop a reliable product it is important to predict effects of, and to deal with uncertainties.

Reliability is an important issue for the products developed by Assembléon, the company where this research project is performed. One of the industries Assembléon serves is the automotive industry. About reliability in this industry the following can be found on the website of Assembléon:
"For the automotive industry, achieving zero defects is a necessity. Legal and safety regulations and the associated high cost of possible field recalls are making product quality and production reliability a number one priority for the automotive industry"
Besides the automotive industry Assembléon serves a lot of different industries that have all one common need for total reliability. That is why reliability is one of the core values of Assembléon.

This focus on reliability has an overlap with the research that is performed at the department of Quality and Reliability Engineering at Eindhoven University of Technology. Within this department Aravindan Balasubramanian is performing his PhD research, with the main objective of estimating and predicting reliability of the product during development and to establish a good reliability diagnostic process to help reliability improvement. To predict the reliability, an Artificial Intelligence technique called Neural Networks is used.
This master thesis is setup as an exploratory research for the PhD project of Aravindan Balasubramanian and therefore has as goal to perform a feasibility study in which is investigated if a neural network can be constructed that can predict the reliability performance of an Assembléon machine.

This master thesis will mainly focus on the collection of data and the construction of a neural network that will enable early reliability comparison. In the next chapter a short description will be given of Assembléon and the products they produce. Further a project description, research questions and the approach of the project is given.

2. Project Environment and Description

This chapter will give an overview of the environment where this project was performed. This will include information about the company and the products they produce. After this overview, a project description is given that will include the research questions that will be investigated. Finally, the project approach will be shortly described.

2.1 Assembléon

The master thesis project is performed at the department 'System Test and Release' (STaR) of Assembléon in Veldhoven. The STaR department is responsible for the verification and validation of Assembléon products. They are also continuously adjusting the way of working to decrease the time to market and keep qualitative products.

Assembléon develops, assembles, markets, and distributes a diverse range of Surface Mount Technology (SMT) placement equipment. With the developed and produced products, Assembléon tries to provide their customers with the most flexible and cost effective means to deal with the fast changing product mix and batch sizes that are common in the electronics industry. Customers of Assembléon use the products to place Surface Mount Devices and a large variety of electronic components on Printed Circuit boards (PCBs).

Assembléon's goal is to contribute towards a seamless manufacturing supply chain, which deals quickly with fast changing market dynamics. Assembléon realizes this by guaranteeing the performance and reliability of their solutions by providing added-value services and support. Assembléon's solutions are easily tailored to meet the speed, flexibility and accuracy the customers need.
Assembléon surface-mount Pick & Place machines are used in production environments ranging from electronics manufacturing to specialized fields like the semiconductor back-end industry and module manufacturing.

Assembléon has a strong global presence with a large network of sales and support offices in Asia-Pacific, USA and Europe. The headquarters as well as the research and development centre are established in Veldhoven, the Netherlands.

2.1.1 Products

Assembléon provides a product range in pick and place robot systems which cover any application and any volume demand in the electronics assembly industry. With the A-Series Platform, Assembléon presents a new blueprint for manufacturing flexibility.
The strength of the A-series is modularity where appropriate, specialization when required. Therefore it looks, feels and behaves as a single machine occupying a small footprint; yet it combines a number of technologies, which enable it to provide the highest pick and place performance for every component to be placed.

Manufacturing an ever-greater variety of electronic goods with an ever more limited shelf life means faster product changeovers, fluctuating volumes and shorter times-to-market. Further there is increasing product complexity and the trend to electronics modules make it all the harder to choose the right line setup to suit all applications. Assembléon helps with the right line setup by offering three different series of products: the A, M and X-series.

Pick & Place solutions require the flexibility and versatility to mix-and-match large or small batches, with a low or high product mix, at increasingly demanding quality levels, while still keeping operating costs low. This is where the A-Series can be the solution.

The A-Series contains the modules AX-201, AX-301 (AX-3) and AX-501 (AX-5). These three modules introduce integrated functionality across a single, common platform. The A-series is most suitable for higher placement rates between 45,000 components per hour (cph) and 150,000 cph. This research focuses on the AX-3 and AX-5, of which a picture is shown in Figure 2.

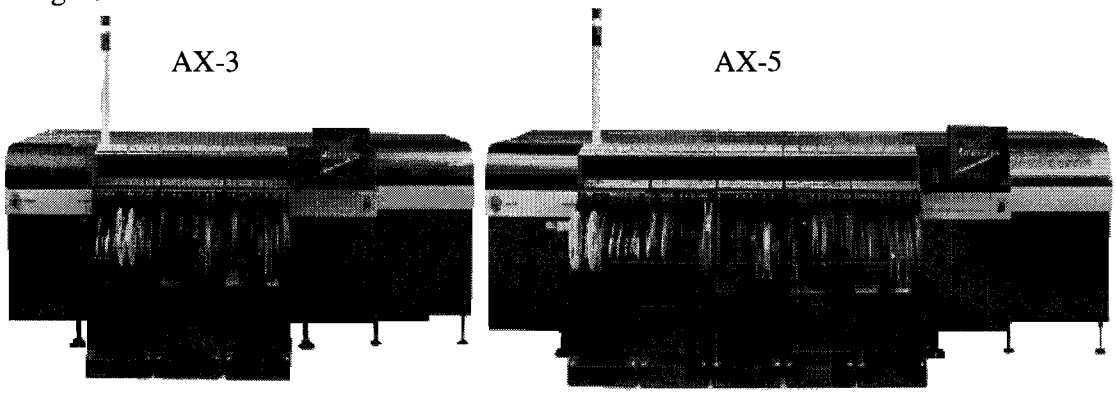


Figure 2: The AX-3 and AX-5

The M-Series places a wide range of components, including chips, complex and odd-form components. M-Series modules (MG-1, MG-2 and MG-8) complement each other for flexibility and optimum productivity. They are optimized for high product-mix and any batch size. Placement rates are up to 40k components per hour.

The X-Series is a cost-effective, Pick & Place platform for low- to medium-volume manufacturing. X-Series modules (Opal-XII, Topaz-XII and Emerald-XII) complement each other to deliver outstanding flexibility and fast ramp-up. Machines are easily optimized for ultra-high feeder count, and large boards. The X-Series has a placement rate up to 20k components per hour.

2.2 Project Description

In modern society software plays an important role. Software is described by Goel [Goe85] as an instrument for transforming a discrete set of inputs into a discrete set of outputs. It comprises of a set of coded statements whose function maybe to evaluate an expression and store the result in a temporary or permanent location, decide which statement to execute next, or to perform input/output operations. Since software is produced by humans, the final product often contains flaws. These flaws are discrepancies between what the software can do versus what the user or the software wants it to do [Goe85].

During the development of software the primary objectives are quality, reliability and customer satisfaction. The process of developing software is a time consuming complex process [Ans99]. The performance of the software development process remains uncertain, because it is hard to determine the performance of the software development process. This is because an important aspect of software testing involves judging how well a series of test inputs, tests a piece of code [Mic01]. The goal of the test is to uncover as many faults as possible with a certain set of test inputs. A series of test inputs that reveals many faults is

obviously better than one that can only uncover a few. It is however almost impossible to predict how many faults will be uncovered by a set of test input data. This is because of the diversity of possible faults themselves and because the concept of a fault is only vaguely defined [Mic01]. This results in uncertainty about the performance of the final software product. When releasing software, the uncertainty should be at a minimum level. Assembléon therefore tests its software before releasing it.

During the development of software the reliability of the software is measured. This helps planning and controlling resources during the development process so that high quality software can be developed. As more faults are exposed by the testing process, the additional costs of identifying the remaining faults rise quickly. Thus, there is a point beyond which continuation of testing to further improve the quality of software can be justified only if such improvement is cost effective. Measuring the software reliability can help determining such a tradeoff [Goe85].
As mentioned earlier, identifying and fixing every failure is time consuming and even after very elaborate testing it is not guaranteed that all failures are found. Therefore Assembléon is searching for a method that can describe in a short period of time whether the new software version performs as reliable as or better than the previous version. With this method Assembléon can determine whether the new software version is ready for release (the reliability is equal or higher than the previous version) or if further development is needed.
John Kerstens [Ker07] performed a research at Assembléon, the goal of his research was to develop a method that determines if a new software version performs equal or better than the older version. Besides developing a methodology to compare reliability, he also tried to identify factors that could explain the large difference in mean time between failures (MTBF) that were measured between different machines within the same customers and between customers. The factors that Kerstens identified were the amount of robots, the number of CV-cameras, and the number of hardware changes. The amount of robots and the number of hardware changes showed a negative relation towards startup failures. The amount of CV-cameras and hardware changes showed a negative relation towards the MTBF.
Since storage/retrieval of the data, that could help to quantify factors, were not optimal earlier, no additional factors were tested. The model that was proposed by Kerstens, to determine whether a new software release performs equal or better, can shortly be described as follows. As described earlier, some of the factors are influencing the reliability of the machines. By keeping these factors constant on the machine when testing two different software releases (the old and new version), makes it possible to compare the MTBF data without having to take into account the influence of these factors. So when a difference in MTBF data is observed between two software versions this is not caused by a difference in values for these factors but by the difference in software version. The research of Kerstens at Assembléon together with the literature study performed in the first part of preparation of this master has resulted in several research questions. The next chapter will contain the research questions and a short motivation of why these research questions are selected.

2.3 Research questions

Kerstens [Ker07] developed a method to determine if a new software version performs equal or better than the older version. The method he developed keeps certain factors, like the amount of placement robots, constant to make it possible to compare the MTBF data without having to take into account the influence of these factors. Kerstens mentions in his research that due to problems with data collection, only few factors have been taken into account. By only taking these few factors into account the difference in MTBF between two tests will probably not only be caused by the difference in software version but also by other, not yet

investigated factors. This is a problem Kerstens also acknowledged. He therefore recommended that the influence of more factors on the MTBF should be investigated, to give a better insight why there are large differences between the MTBF of different machines. These factors should also be incorporated in Kerstens or another model to get better comparison of the influence of the software version on the MTBF, and thereby be more certain about whether to release the new software version or not.
The recommendation of Kerstens indicate that the first step in this research should be the investigation of which factors have a large influence on the reliability of pick and place machines developed by Assembléon. These factors should in the next step be incorporated in the model that is used for the reliability comparison at Assembléon. This leads to the first research question of this master thesis project.

1. Which operational conditions (factors) should be included in the model to give a good representation of the real situation?

- Investigating which operational conditions are of importance to give a good reliability prediction with the model.
- Investigate the practical usefulness (e.g. is there enough data available) of the selected parameters.

The method developed by Kerstens keeps the value of the factors of influence (i.e. the operational conditions) constant between the tests of two succeeding software versions. This will eliminate the influence of these factors on the difference in MTBF data of the machine caused by two different software versions.
A part of the reliability testing is done at reference sites, these are selected customer sites that run early prerelease software versions and provide feedback about these prerelease software versions. At these reference sites, the systems updated with new software version are tested for their Mean time between failures (MTBF). The reference sites are production environments, so it is not possible to create the same conditions when testing both software versions. Another possibility would be to let a machine run at Assembléon that would run under the same condition but under the old software version. The problem with this method is that;

1) Assembléons test lab is not setup as a production environment.
2) When producing board, components are placed on the PCB. The costs of the total amount of components needed during the test would be high because a large amount of boards would be produced during the test period.
3) When producing boards, an operator has to be present to solve possible problems. The test period required for getting reliable results would cause high operator costs.

Not being able to compare the machine with two different software versions under the same conditions makes it impossible to use Kerstens model to determine the influence of different software versions on the MTBF of the machine. Therefore the application of other prediction methods has to be investigated, to determine their usefulness in determining the reliability of the machine under different operational conditions.
The literature review in the first part of the preparation [Fra08] shows that classical prediction methods like a regression model have limitations that make them less useful. Some of these limitations are:

- Classical methods are incapable of handling data with high uncertainty properly.
- To use classical methods, assumptions have to be made about the type of relationships that exist between parameters, because these relationships are not known upfront it is preferable to use a method in which no assumptions have to be made about the type of relationships existing between parameters.

Other prediction methods like Bayesian frameworks or neural networks (NN) could be a solution. When using a Bayesian framework, the input for the model has to be gathered in the form of prior distributions. For the use of neural networks a large data set has to be available in order to train the neural network.
Assembléon's products are equipped with software that registers a large amount of different parameters that represent the machine configuration, product characteristics, operation characteristics, and machine performance.
Because this information is registered/logged automatically a lot of data points could be made available over time, this makes the use of neural networks probably possible. Gathering prior distributions of all the different parameters is more time consuming because these are not automatically gathered by the machine. This makes the use of Bayesian less practical.

From the literature, neural networks seem to be a good candidate to use as reliability prediction method in the case of Assembléon. This because no prior assumptions have to be made about the relationships that exist between parameters, and NN are capable of handling uncertain data. Next to these advantages there is also a disadvantage in using a neural network for predicting the reliability of the machines developed by Assembléon. The information that is registered by the machine makes it possible to use neural networks as just mentioned. The collection of data by the machine means that the software version has to be in an advanced state of its development in order to run on a machine. The software has to be in its test phase in order to collect data about the performance. Predicting the reliability of a product early in the development process is of importance in order to make changes to the design as early as possible, which reduces time to market and costs. Because machine logging information is only available from the moment in time the test phase of the software development process starts, it is not possible to develop a neural network that could predict the performance of the new software version earlier in the development process.

This is however not a disadvantage that is specific to neural network. All prediction techniques that need a reasonable amount of data, are probably not suitable for predict the performance of a new software version earlier in the development process. For instance a linear regression model requires according to Hair [Hai05] a minimum of 5 observations for each independent variable, but preferably 10 tot 20 observations for each independent variable. These amounts of observations are also not available during earlier stages of the development process.

Besides that from the literature, a neural network seems to be a good candidate to use as reliability prediction method in the case of Assembléon. The exploratory nature of this research, as described in the introduction, also steers in the direction of using neural network for reliability prediction method.

The neural network will be used to model the relation between usages and configuration of the machines at customer's sites and their failure data like the mean time between failures. By modeling this relation it should be possible to estimate the MTBF of a machine according to several configuration parameters and operational conditions.

By testing new software versions at a reference site and producing boards until failures occur it is possible to determine the MTBF of the new software versions. As mentioned earlier it is not possible to use reference sites or test facilities at Assembléon to determine whether new software versions are ready for release, because configuration and operational conditions can

not be kept constant. The model (NN) can estimate the MTBF of a machine by several configuration parameters and operational conditions. By determining the MTBF of the new software version, by testing at the reference site, and simulating the same conditions, but with an earlier software version, by modeling it in the neural network it is possible to estimate whether the new software version performs equal or better than earlier versions. Figure 3 graphically displays the method to determine whether a new software version is ready for release.

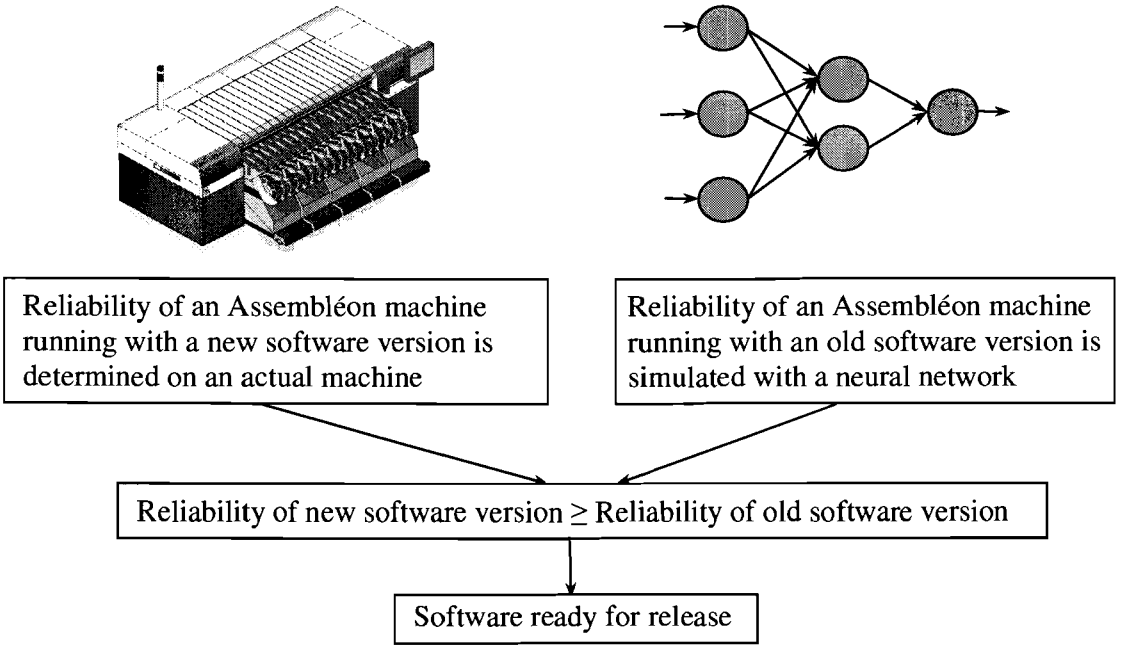


Figure 3: graphical representation of the method to determine whether a new software version is ready for release

The literature that was studied on neural networks did not give clear guidelines about choosing the configuration and size of a neural network. This gap in the literature makes it hard to determine the amount of data that should be gathered and how many factors can be taken into account in the neural network that will be developed for Assembléon. This results in the following two research questions

2. What influence does the amount of (training) data have on the accuracy of the neural network?

3. What influence does the addition of extra input factors have on the accuracy of the neural network?

The previous two research questions are related to the implementation of a neural network that should predict the reliability of machine developed by Assembléon. The last research question is more in line with the goal of Assembléon to develop a method that can describe in a short period of time whether a new release performs as reliable as or better than the old and existing release. So the fourth research question is

4. Can the neural network be used to compare different software releases under different operational conditions?

The project approach that will be described in the following section gives an indication of how this exploratory research is setup in order to find answers to the just stated research questions.

2.4 Project approach

The four research questions discussed in section 2.3 specify the overall target of this thesis project. In this paragraph a short description will be given on the overall project approach that is used as a guide throughout the project in order to answer the stated research questions.

A literature study is performed to get a good insight in the basic principles of reliability estimation and the prediction technique neural networks. Chapter 3 summarizes the literature that is used as a basis for the further research.

After the summary of the literature study in chapter 3, chapter 4 describes the different applications of a user profile and the process of creating a user profile for Assembléon which includes among other things brainstorming and data collection. The user profile will be used as basis for the data set needed for the training and validation of the neural network.

Chapter 5 presents the process of transforming the user profile into useful input for the neural network. This process includes reducing the amount of input variables and coding values. This chapter also describes the process of creating a neural network that is used for modeling the relation between usages and configuration of Assembléon's pick and place machines and their the mean time between failure. This chapter will help to answer the four research questions stated in section 2.3.

The construction of a linear regression model, that helps to evaluate the performance of the neural network that is developed in chapter 5, is described in chapter 6.

Chapter 7 will present the conclusions of this research and will therefore get back to the research questions.

Chapter 8 will discuss the problems encountered during the exploratory research and the possible solution to these problems. To solve some of the problems more research is needed and for other problems better, automated data collection has to be realized.

In chapter 9 some recommendations will be given that could be used for future research.

3. Literature

As preparation to the master thesis, a literature study is performed. This literature study dealt with different subjects related to reliability prediction. This literature study formed the basis for the research that is performed during the master thesis. Important aspects of the literature study, in relation to reliability prediction and neural networks will be summarized in this chapter.

3.1 Reliability

Reliability is an important attribute of a product because failure of the product can be very costly and lead to unsafe situations. Generally reliability is defined as "the ability of a product to perform a required function under stated conditions for a stated period of time" [Wal06]. This definition can be translated into a probability measure that is supported by models such as reliability block diagrams and fault trees.
Inaccurate reliability prediction can result in the development of an unreliable product as result of optimistic predictions or overdesign. The development of unreliable products can have many consequences. If the reliability prediction is too optimistic this can cause disasters like the U.S. Shuttle failure. And if it is too pessimistic, this could lead to overstocking of spares and waste of money [Won90].

There are at the moment four major categories in which reliability methods can be classified; reliability prediction, qualitative methods, quantitative methods, and analytical methods. Prediction methods are based on data-bases like MH-217. The larger the database the more accurate prediction can be made, because more detailed information (failure rates) about different components is available. When information about more components is available a system can be modeled closer to the reality. These models are mainly used to establish a baseline reliability figure while the design is still on paper [Eco04]. Quantitative methods mainly make use of computer simulation analysis to predict the reliability, and are therefore computationally intensive and time consuming. Qualitative methods use accelerated testing environment to subject the product to elevated stresses in order to determine failures or design weaknesses. Analytical methods combine the three earlier described methods. It is a mixture of prediction and quantitative methods but it requires information obtained from qualitative methods. The quality of the results of these methods depends on how they are used, how the results are interpreted, and what actions are taken based on the results [Eco04].

Reliability prediction tools often use complex mathematical models to predict the reliability of systems. These models that are used in the prediction tools are abstractions of the design. The mathematics that is used in the models may not be able to represent the behavior or structure of the design in the correct manner [Lee89].

Besides limitations in the prediction methods itself there are also problems with translating input and output for the model to reality. As mentioned in the section above, different input parameters can be used to determine or predict the reliability. It is not always clear which input parameters to use to represent the reality as close as possible and it is not always possible to quantify certain parameters directly. For example, the parameter user skill is hard to quantify because it does not only depend on years of experience but also on education level.
Translating the results of the model to the reality can also cause some problems. The models are abstractions of the reality, so when interpreting and evaluating the results of these models, it should be taken into account that this is not a precise representation of the reality.

Another limitation of classical prediction methods is the inability to deal with uncertainty. The reliability of a product can be expressed in the functional calculation of the reliability (point estimate value) and the uncertainty (usually represented by a distribution function) that accompanies that reliability value [Ker98]. Classical methods don't incorporate uncertainty when using reliability numbers [Siu98]. Information that is used to predict reliability early in the development often contains a lot of uncertainty. Not incorporating this uncertainty into the reliability prediction will result in incomplete estimations of the reliability.

3.2 Uncertainty

The data that is collected for a reliability prediction contains uncertainties. The earlier the assessment of reliability data, the greater the degree of uncertainty in the data. The degree of uncertainty can decrease during the development process as and when more information becomes available. But on the other hand, the uncertainty can also increase because of more design alternatives available. For example in the conceptual design different concepts are developed and each model introduces different uncertainties. It is important to quantify the degree of uncertainty of data, in order to take the uncertainty into account in the reliability prediction [Kap90]. In addition to uncertainty in data, there are more factors that influence the accuracy of reliability prediction. Some of these factors are:

- The accuracy of the mathematical model.
- The presence or absence of gross over-stressing during operation.
- The selection of the correct and relevant failure rate data.
- The elicitation of biased judgments from experts.

This last factor can have large influence on the accuracy of the reliability prediction because early in the development process not much information is available except for the experts' opinions about design concepts.

After discussing the limitations of classic prediction methods, next a prediction methods will be discussed that possibly can overcome the limitations described earlier. As described in section 2.3 neural network could be a good candidate to use as reliability prediction method in the case of Assembléon. Therefore this method will be discussed in the next section.

3.3 Neural network

Neural Networks (NNs) are software, or hardware based constructs that use algorithms that are capable of performing functional input/output mapping. NNs can be used as a tool for prediction and are used in several different fields. NN are used by Jones [Jon02] for predicting the reliability of an aeronautical engine. Dirusso [Dir02] uses it for predict pediatric trauma death and Papadrakakis [Pap96] uses an NN to predict structural reliability.
NNs are composed of several basic units (neurons) that can loosely be compared to neurons in the brain. The neurons are linked together by weighted connections (synapses) that can change their strength. Each neuron can independently integrate information that is provided by the weighted connection and process this information into a response [Abd94].
NNs are especially useful when the physical–mathematical models to be described by them, are complicated or unknown. This is because NNs acquire knowledge through learning. During the learning process the strength of the weighted connection between neurons are determined. This learning process makes it possible to use this tool to model complex systems for which mathematical descriptions are not available [Raj05]. Correctly training a neural network is important and will therefore be discussed a bit more elaborate in section 3.3.3.

The development of neural networks has started around 1940 but there were little developments until the late fifties and early sixties when Rosenblatt [Ros58] introduced the perceptron. This model possesses almost all the essential features that now-a-day neural networks possess [Abd94]. Perceptrons have two layers of neurons, the input layer and an output layer. They were mainly used to make associations between a binary input and output. The state of the input neurons (0 or 1) and their weighted connection with the output neurons determine the response of the output neurons (0 or 1). The learning process determines the weight of the connections between the neurons in the two different layers. The problem of these early models was that it was only possible to train the model when there was a linear transformation possible between the input and output.
In order to overcome the problem of only being able to use linear transformation between the input and output, hidden layers were introduced. At first, this was not as promising as thought, because there were no training rules for neural networks that contained hidden layers. These training rules were developed over time and will be discussed in a later stage [Abd94].
In the description of the history of the development of neural networks above the term input, output and hidden layer are addressed in the following section these terms and the architecture of neural networks will be described.

3.3.1 Layers in a Neural Network

Nowadays Neural Networks are often composed out of three different kinds of layers, the input layer, the hidden layer, and the output layer. Figure 4 shows an example of a neural network that is composed out of three layers.

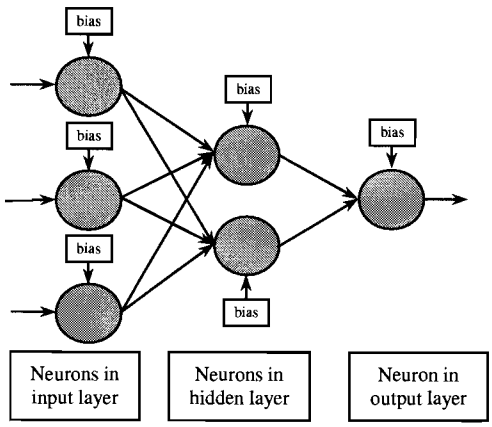


Figure 4: Example of a 3-layer neural network [Raj05]

The three different kinds of layers can be characterized as follows [Raj05]:

Input layer

The first layer is called the input layer. It can consist of several neurons that are each linked to a specific input parameter by a weighted connection. Each neuron also has an associated bias that modifies its output. This bias can be compared to the intercept term, the constant term in the regression equation, found in a regression model [Tu96]. The intercept represents the value of the dependent variable when all independent variables have a value of zero. In a neural network the bias has the same function. When all inputs to a neuron are zero the transfer function can have any input under the same conditions by learning an appropriate value for the bias. The bias is much like a weight; the strength of the connection with the

neuron will change as result of training. The difference between a weight and a bias is that the bias has a constant input of 1, while a weight has a variable input as result of the output of the neuron to which it is connected. The number of neurons in the input layer can be varied in order to meet the requirements of the particular application.

Hidden layer
The next layer(s) are the hidden layers. These layers consist of a set of neurons that are also linked to the preceding layer by weighted connections. Each neuron also has an associated bias that modifies its output. A Neural network can consist of more than one hidden layer. Each of these layers may be constructed out of different numbers of neurons. The amount of hidden layers depends on the complexity of the situation that has to be modeled. Literature about neural networks doesn't propose a manner or rule that can be used to determine the amount of hidden layers and the amount of neurons in each layer. Demuth [Dem07] states that a larger numbers of neurons in the hidden layer give the network more flexibility because the network has more parameters it can optimize. If the hidden layer is made too large, the network might cause the problem to be under-characterized. But Demuth does not propose a specific manner, except from trial & error, to determine the optimal amount of layers and the amount of neurons in these layers.

Output layer
The last layer is the output layer. This layer can contain a number of neurons that equal the number of relevant outputs or less. The neurons in the output layer are connected to the neurons in the preceding hidden layer by weighted connections. The neurons in this layer also contain a bias that modifies its output.

Each neuron of a neural network has a transfer function that determines its output. The purpose and some examples of this transfer function are given below.

3.3.2 Transfer function

The transfer function that is associated with each neuron determines the output of the neuron. Before a transfer function determines the output of a neuron, first the activation of the neuron is computed as the weighted sum of all inputs (including the bias) of the neuron. Some commonly used functions are [Raj05], [Dem07]:

- Log-Sigmoid transfer function (Logsig): this function takes input values between plus and minus infinity and converges the output into the range between 0 and 1.
- Tan-Sigmoid transfer function (Tansig): this transfer function transfers the input, which can vary from plus to minus infinity, into an output between -1 and 1
- A linear function (Purelin): this function does not perform any action because its output is equal to the input. A Purelin is usually used at the output stage of the neural network. The output stage often doesn't need to transform its output into a range between 0 and 1. This is because the output of the neural network represents a parameter (e.g. failure rate) that has value range that is bigger than the range between 0 and 1.

Each neuron can have a different transfer function but in general, the neurons in a specific layer use the same transfer function. Rajpal [Raj05] uses trial and error in his research to determine the correct transfer functions for the different neurons, while Anderson [And95] states that in almost all the cases a Log-Sigmoid transfer function should be used. In the literature there are no specific guidelines given which transfer function to use in which situation. Demuth [Dem07] states, that the best architecture to be used for the network depends on the type of problem to be represented by the network.

After describing the different layers of the neural network and the functionality of neurons, in the following section the training of the neural network will be discussed.

3.3.3 Training the NN

The training of a neural network works, as mentioned earlier, by adjusting weighted connections between neurons so that a particular input leads to a specific target output. Figure 5 shows a graphical representation of a learning process of a neural network. The output of the neural network is compared to a target value; if there is a difference between target and the output value an adjustment is made to the weighted connections between neurons in the neural network [Dem07].

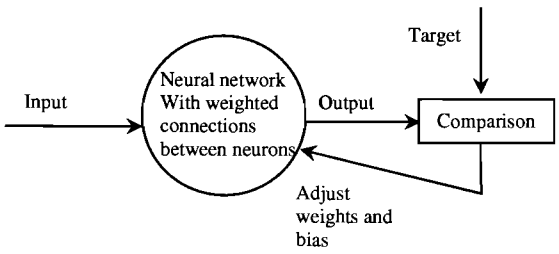


Figure 5: Training of a neural network [Dem07]

The training that is described above is called perceptron learning rule. A learning rule is defined by Demuth [Dem07] as a procedure for modifying the weights and (associated) biases of a network. This procedure can also be referred to as a training algorithm. The perceptron learning rule is a basic learning rule that was developed in the early sixties. One of the disadvantages of the perceptrons was that they only learned when the final state was wrong.

The purpose of the learning rule is to reduce the error between target and output value by adjusting the weighted connections and biases. Widrow and Hoff [Wid60] found that it was possible to determine the direction towards the minimum of the error term. This resulted in the Widrow-Hoff learning algorithm reducing the learning time and the sample size of the required training data [And95].

The training algorithm described above has a large disadvantage. It can only be used when the neural network consists of only an input and output layer. As soon as hidden layers are added these algorithm can't be used because it can't identify the amount of influence a neuron in the hidden layer has on the difference between actual and assigned output. Therefore the training algorithm can not adjust the weights of the neurons in the hidden layer. The backpropagation learning algorithm was developed to overcome this problem. The backpropagation algorithm is one of the most used learning algorithms and the basis is the same as the Widrow-Hoff algorithm. It calculates the errors of different neurons in a layer. These errors are then summed and propagated backwards towards an earlier layer. The neurons in the hidden layer then estimate their own error as a function (e.g., a weighted average) of the error of the units in the subsequent layer. This backward propagation makes it possible to train hidden layers [Abd03].

3.3.4 Merits & demerits of neural networks

The backpropagation learning rule and the structure of the neural network make it possible for NNs to have nonlinear relations between the independent (input) and dependent variables (output). This is one of the advantages of neural networks over standard statistical approaches. Another advantage is that NNs allow all possible interactions between the dependent variables. To allow this flexibility, standard statistical approaches (e.g., logistic regression) require additional modeling. In addition, Neural networks do not require explicit distributional assumptions (such as normality) in order to handle uncertain inputs.

Apart form the advantages of NNs there are also some drawbacks. The primary disadvantage of an NN is its "black box" quality. Without extra effort, it is difficult to gain insight into a problem based on an NN model. Regression techniques allow for example the user to stepwise eliminate possible explanatory variables that do not contribute to the fit of the model. This could also be achieved with neural networks, but this would be very time consuming because different models have to be developed with different combinations of variables. Regression techniques also allow hypothesis testing regarding both the univariate and multivariate association between each independent variable and the dependent variable of interest. These are the features that are not standard available for NNs [Sar01]. Another disadvantage is the large data set that is required to train and validate the neural network. As a rule of thumb some authors say that for each weight there should be 10 data points [Zha98]. Figure 4 shows a simple neural network with three input neurons, two hidden neurons and one output neuron. The network contains 8 connections (weight), so for the training of this small network already 80 data points have to be collected. When adding one neuron to the input layer and one neuron to the hidden layer the network would have 15 connections, which requires 150 data points.

4. User Profile

The literature of the previous sections has given an indication of the different aspects of a neural network and how it works also advantages and disadvantages of the method are described. An important part of creating a neural network is creating a data set that can train and validate the network. Before a data set can be created it is important to determine which variables / factors should be included in the neural network, so the network can give an accurate prediction of the reality. Within Assembléon this data set could also be used for other purposes as will be described in the next section.

4.1 Creating a user profile

The collection of several indicators that represent how a product/system is used by its user is referred to as a user profile. Within Assembléon they are not only interested in how users use their machines but also what kind of machine configuration is used, which software version is used, the performance of the machine, and what characteristics the printed circuit boards have that are produced with Assembléon machines. All these different indicators will be incorporated in a specific profile that is created for each machine for which data is available and this profile will be from now on referred to as a user profile.

The collection of user information in a user profile can be valuable in various stages of the product life cycle of Assembléon machines. During the development of a new machine it is useful to know how and who used earlier machines. This helps with the defining which (new) features a machine should have and can help to make machine more intuitive in their usage. When a machine is in a later stage of the development and prototypes become available, the user profile can be used to setup test programs. The placement machines of Assembléon contain many different functionalities and the time to market is getting shorter for each new design. To test a machine in a short as possible time it is important to know which functions are used most often, so test programs can focus on these functionalities. The user profile can provide this information. When a machine is released to the market, information about the usage of the machine should be collected to provide information that can be used service purposes and to determine whether the machine performs as expected.
The user profile can provide valuable information for failure analysis. When root causes of failures can be determined, this could be used as input for improvements. For the purpose of failure analysis, Assembléon initially started collecting data about how their systems are used by their customers.
The user profile in this research will be used as input for the neural network. By gathering user profiles from a lot of different machines a data set is created that can be used for the training and validation of the neural network. The process of using the user profile as input for the neural network will be discussed in the next chapter.

A lot of information can be collected about the properties of a customer (e.g. where they are located, how many Assembléon machines are present, and how many operators are employed) and how a machine is used (e.g. how many placement programs are run, the type system controller that is used, and the temperature of the placement head). Collecting and analyzing all the available information would be very time consuming, it would also provide a lot of useless information. Determining which data should be collected is hard because it is not clear which information is relevant to collect. Which data is relevant partly depends on, for which purpose the user profile is used. When the user profile is used for testing purposes it could be that other factors are more relevant than when the user profile is used for

development purposes. In this research the focus lies with reliability prediction, therefore the user profile focuses on factors that influence the reliability of the placement machine. The first step in setting up a user profile for Assembléon was determining which information to include in the user profile.

4.1.1 Brainstorming

To get a better idea of which information to include in the user profile a brainstorm session was organized. The results of this brainstorm session are presented in Figure 6, a larger version of the figure is included in Appendix A. The figure shows that there are 4 main categories; Operation conditions, machine performance, machine configuration, and product characteristics. Each of these main categories, except for product characteristics, is subdivided in several different sub-categories like changes, efficiency and software. These sub-categories and the main category product characteristics are split up into separate factors.

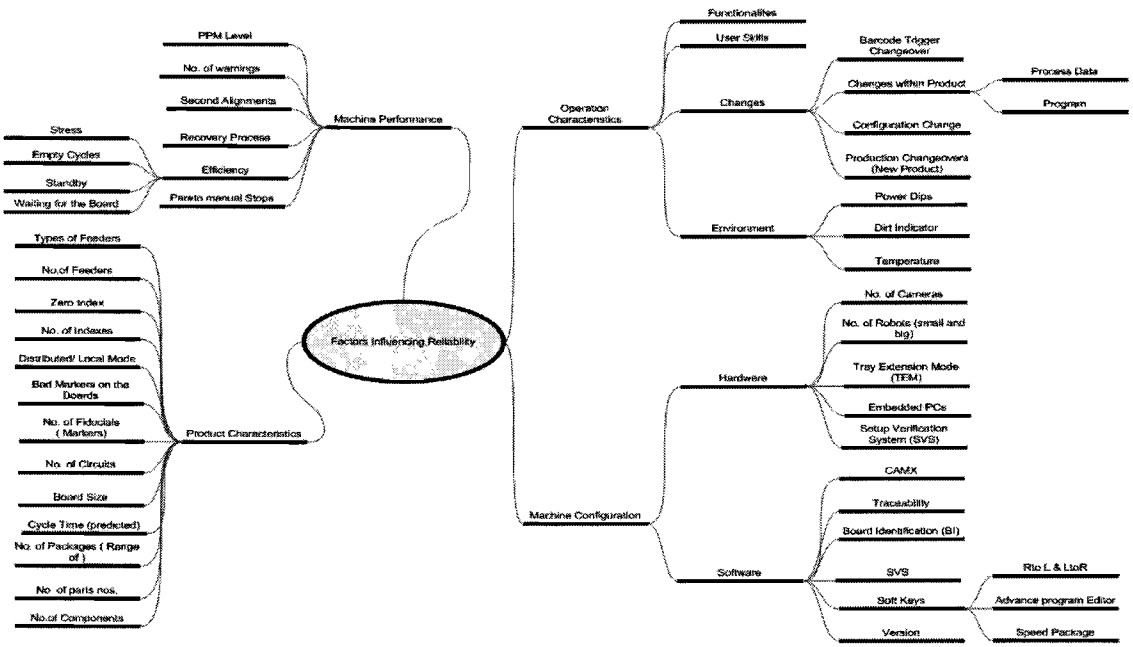


Figure 6: results of the brainstorm session

After this brainstorm session a questionnaire was developed that used a 1 to 7 likert-scale. The questionnaire was used to determine how significant a certain factor influences the reliability of the system, according to Assembléon employees. The questionnaire was distributed under 70 employees of Assembléon. The response rate of the questionnaire was 60%. The relative weight method [Bal08] was used to create normalized ratings to remove biases. The relative weight method normalizes the rating (r) of the ith factor given by the jth respondent (j) by dividing the rating by the summation all the ratings of respondent (j) on all n factors.

$$N_{ij} = \frac{r_{ij}}{\sum_{i=1}^{n} r_{ij}}$$

The biases that are removed by normalizing the ratings could occur due to the fact that respondents give bias scores to factors from their functional view. The overall rating for each factor was determined by averaging all the normalized ratings. The higher the score, the more

significant the influence a factor has on the reliability of the system, according to the employees of Assembléon. The list of factors in combination with their relative weights is presented in Appendix B.
This list of factors formed the basis for the user profile. The next step in creating the user profile was determining of which factors data could be collected. For a majority of the factors logging data could be used to collect information about the factor. The different sources of logging data will be discussed in section 4.2. For a few parameters customer information was available within Assembléon (e.g. the region in which the customer was located). For several other factors no direct information could be collected (e.g. user skill) and therefore are excluded from the user profile for now. Eventually these factors could be included but then an indirect measure should be constructed. This was beyond the scope of this research.
During the first exploration of the different data sources it became clear that of the four main categories that were identified, during the brainstorming session, the factors subcategorized under main category operation characteristics were hard to quantify. Several factors, like functionalities, can be quantifying by analyzing the log data that registers the use of the graphical user interface (GUI). This process has to be performed manually and is therefore very time consuming. Because this is an exploratory research and of time restrictions it was decided to exclude these factors even though the excluded factors could have a significant influence on the of the reliability of the system, which influences the performance of the model that is going to be developed. The exclusion of these factors results in that only the following four factors (Table 1) could be collected for the main category operation characteristics.

Table 1: factors subcategorized under main category operation characteristics

Sub-category	Factor
# of CO's (per week/month)	# programs run / week
	# of different programs run / week
(Environmental) temperature	Average XY Temperature
	Average PH Temperature

Appendix C shows the different subcategories with their underlying factors of which data could be collected and are therefore incorporated in the user profile.

4.2 Data collection

After determining which factors could be included in the user profile from a practical point of view, a start could be made with collecting the data from different customers. The data collected from the different customers contains sensitive information from a competitive point of view, customer and machine names are altered in this report.

Information used for construction a user profile is gathered from different sources from customers of Assembléon that are allocated all over the world.

There are different data sources used to gather information for the user profile. These sources are TBF data, Eventlog, RAM data, and placement program. Each of these different data sources is shortly described on the next page.

TBF data:
TBF data registers the Time between failure data. This service logs information like the configuration of the machine and the time the startup of the machine takes place. Next to

startup data also shutdown data is logged. In this shutdown log the time the machine is shutdown is logged and the last 5 errors that occurred are logged in order to be able to determine what the cause was of the machine shutdown. The information in this data source is used as basis for the calculation of the mean time between failures, which is used as reliability measure in this research.

Eventlog:
The eventlog registers different kinds of events, like errors that occur, process information and actions that are taken in the graphical user interface. This information can be used, besides as input for the user profile, for root cause analysis.

RAM Data:
Reliability Availability Maintainability data. The RAM data is logging information that registers all the performance indicators of the machine. Some of these performance indicators are productive time and the pick performance. This information can be used for Reliability Availability and Maintainability purposes.

Placement program:
The placement program (pp) is a file that contains all the information needed to produce a certain type of board, it contains information about the dimensions of the board, where which component have to be placed, which toolbits should be used for a specific placement action and much more.

4.2.1. Data gathered for the creation of the user profile

The data that was used to construct the different user profiles was collected from 8 different customers with a total of 31 unique machines. Data was collected over three periods of time for 3 machines of one customer and two periods for 6 machines of two different customers. For 21 machines only data for one period was collected. The selection of the customers, machines and the amount of time periods in which data could be collected are based on availability. When data became available for a specific machine of a customer, a user profile for this machine was created.

Figure 7 graphically displays the distribution of the data.

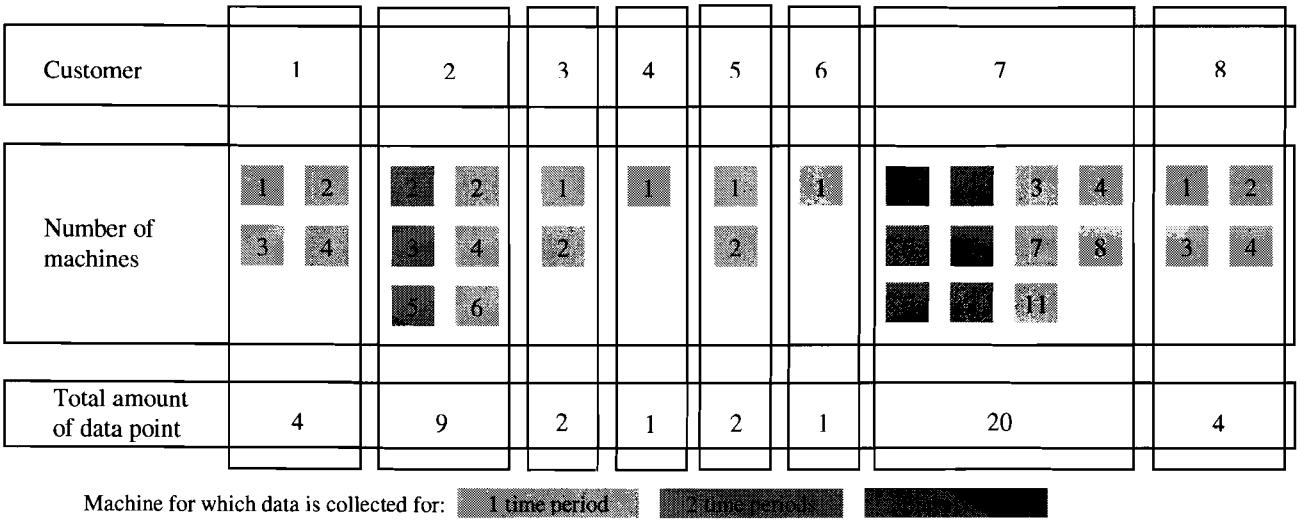


Figure 7: graphical representation of the distribution of data over the different customers

The total amount of 43 user profiles is collected. For some of these 43 user profiles not all data sources were available, which implies that no information is available for several factors. For 3 machines the machine configuration could not be determined.

The amount of 43 user profiles is a small sample size when using neural networks but at the moment this research was performed more data could not be collected. A possibility to increase the sample size would be to select a smaller time period in which data was collected. The time period used to collect information was for most of the machines about one month. This period was party selected because some of the information incorporated in the used profile was already gathered in a one month period at the start of the research. Another reason was the choice of using the MTBF as reliability measure. To determine an MTBF, failures have to occur. When information would be collected in a short time period of for instance one week no failures could have occurred which made the calculation of an MTBF impossible. The collection of information over for instance 6 months would make it possible to calculate a MTBF for the machine. This large time period would on the other hand make the use of some factors useless. When trying to identify the relation between certain factors and the MTBF it is important that the data is collected in such a time period that this relation can be found. When for instance the relation between amounts of placement programs run and the MTBF is investigated, counting the amount of placement programs run in 6 months wouldn't be very useful. A large increase in the amount of placement programs run in a short period of time could maybe cause failures within the machine, which decreases the MTBF. This large increase in placement programs would not be noticed because only the total amount of placement programs run in 6 months is known. This would make it impossible to identify the relation (when present) between number of placement programs and the MTBF.

The user profiles that are collected are grouped together in one file, which from now on will be referred to as the user file. Over time the user file can be extended with more user profiles when more data of (different) machines becomes available. Also more factors can be included in the user profile. Before more factors can be included in the different user profiles, a research has to be performed to investigate which data can be collected to represent these factors in a reliable manner.

After all the data is collected to construct the user profile, as described in this section, the first steps in constructing a neural network can be taken. The construction of the neural network is described in the next chapter.

5. Construction of a neural network

The data that is gathered, on the different factors, in the user file will be used as input for the training of the neural network. Before the neural network could be created several steps have to be taken. The first step is defining the problem for which the neural network could give the solution.

5.1 Problem definition

The development of a neural network starts with getting more insight into what the neural network is going to represent. Section 2.2 described the method developed by Kerstens [Ker07] and also made clear that due to the testing of early software versions at production sites makes it impossible, as Kerstens proposed, to keep the value of the factors of influence constant between the tests of two succeeding software versions. Thereby determining if the new software version would make the machine more, or as reliable as the previous software version.
The predictive performance of the neural network might make it possible to estimate the reliability of a machine by several configuration parameters and operational conditions. To compare the reliability performance of two software versions, both versions should be tested under the same conditions (configuration, operational conditions). One version will be tested on a machine at a reference site, this will be the "new" software version. The neural network will be used to estimate the reliability performance of a machine that runs under the same conditions as the machine at the reference site, but on an "old" software version. This method could make it possible to give an indication if the new software version performs equal or better that the old version. The problem for which the neural network will be used is:

Assembléon is searching for a method that can describe in a short period of time whether a new software version performs as reliable as or better than its previous version.

For the problem described above, a neural network could be the solution. But the problem definition for the neural network would be:

Predicting the reliability performance of an Assembléon machine, by using values of operational conditions, machine configuration, machine performance, and product characteristic.

5.2 Data collection

In chapter 4 a description is given of how a user profile can be created. The values of the different factors included in a user profile will form the basis for the input of the neural network. As described in section 3.3.4, a disadvantage of the neural network is the large amount of data that is needed to train the neural network. Because of the time restrictions of the graduation project, only 43 user profiles could be collected. This amount of user profiles restricts the size (the amount of input factors) the neural network can have. To be able to train the neural network in a sufficient manner the amount of factors included in the user profile is too large. So not all the factors included in the user profile can be included in the training data set. To reduce the amount of factors, an evaluation of the different factors is made. In this evaluation a deliberation is made why a factor should or should not be included in the training data set. In the next section the factors are described that will be used as input for the neural network. Section 5.3.6 will describe the output factor of the neural network.

5.3 Evaluation of factors in the user profile

The evaluation is performed per main category starting with the machine configuration.

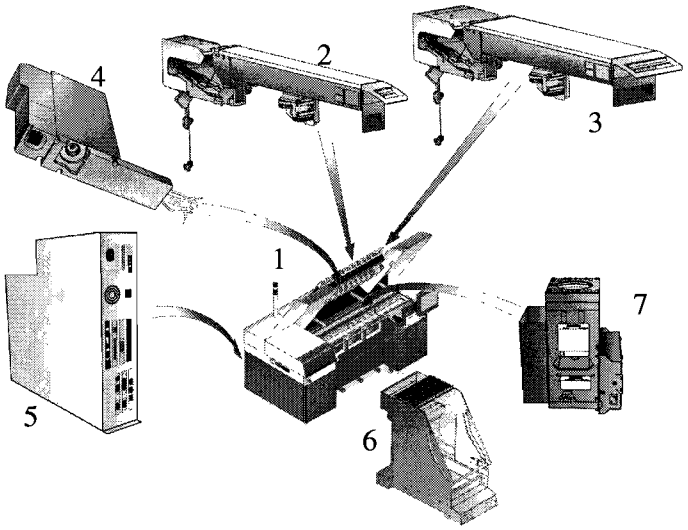
5.3.1 Machine configuration

The factors of which information can be collected in the main category machine configuration are shown in Table 2

Table 2: factors of the machine configuration included in the user profile

Sub-category	Factor
Installation	Region
Hardware configuration detailed	Base type (AX3/AX5)
	# of placement modules
	# of CPR's / SPR's
	# of empty robot places
	# of Prodrive / Advantech PC's
	# of PH-LV / PH-SV
Hardware / software configuration, options	Tray present (y/n)
	BI enabled (y/n)
	Transport direction (L2R or R2L)
	SVS present (y/n)
	Speedpack 1 / 2
	Application SW version

Some of the factors are related to each other. When factors are strongly related, one can indirectly represent the other factor. This makes it possible to delete one of these factors from the training data set. To get a better idea of what the factors represent, that are included in the main category machine configuration, Figure 8 shows several components of the placement machine that are included as factor in the main category machine configuration.



1. Base type
2. Compact Placement Robot (CPR)
3. Standard Placement Robot (SPR)
4. Placement Head Laser Vision (PH-LV)
5. Placement Controller
6. Feeder Trolley
7. Component Vision (CV) Camera

Figure 8: Components of a placement machine that are included as factor in the main category machine configuration

The **region** represents the region in which the machine is placed. Region should be included in the data set because as will be discussed in section 5.3.4 the region in which the machine is placed can have indirect influence on the average recovery time. Also influences like humidity or environmental temperature can not directly be measured but could maybe partly be represented by the region in which the machine is placed. In this research the region is expressed in the continent in which the machine is placed.

The **base type** should be used because this gives an indication of how many robots can be placed on the machine. An AX3 model can contain 12 compact or 6 standard robots, while an AX5 can contain 20 compact or 10 standard robots.

The amount of compact and standard robots (**# of CPR's / SPR's**) in combination with the base type make it possible to delete the factors # **of placement modules** and # **of empty robot places,** because with the amount of standard and compact robots added up together. The amount of placement modules is known and in combination with the base type, which provides the maximum number of placement robots, the amount of empty places can be determined.

Each robot makes use of a placement controller. This is a PC that controls the actions performed by the robot. In the data set used in this research, the controller PC's are made by two different companies, named Prodrive and Advantec. Because controllers are made by two different companies there is a possibility that one company produces a more reliable controller than the other, therefore the distinction between whether a controller is produced by Prodrive or Advantec should be taken into account as a factor. Because, in the data set available, all the controllers in one machine are produced by the same producer, it is not necessary to count the amount of controllers made by a certain producer but it is only necessary to mention which producer made all the controllers in one machine.

All machines in the data set use **placement head laser vision (PH-LV)**, don't have a **tray present, BI disabled** and use a **transport** direction Left to right (L2R). Because these factors are constant for all the machines in this data set, they can be left out of account for the neural network.

The presence of **Setup Verification System (SVS)** should be included. SVS is used to check whether the feeders have been loaded with correct set of components. If this is enabled the chance of placing a wrong component is very small. Therefore the amount of failures related to wrong placed component should be small.

The **speedpack** is a software option that enables the machine to have a higher output. There are two speedpacks available. The first speedpack only enables some software settings to increase the output of the machine. The second speedpack enables the robot arms to move faster, this speedpack also requires the placement of extra ventilators to deal with the extra heat development. The higher output and extra heat could cause more failures therefore the type of speedpack used should be included in the model.

The neural network that is being developed will be used to determine whether a new software version is ready for release. Therefore a comparison is made between the new software version and its predecessor. Including only machines running on the predecessor software version in the training set of the neural network would be desirable because then the improvement of the new software version would be tested against its predecessor. Only using

the information of machines running on a predecessor software version would not generate a large enough sample size to train the neural network, therefore information of machines running on older software versions are included. The factor **software version** is therefore included in the data set.

Table 3 summarizes the factors, which are sub-categorized under the main category machine configuration, that are included in the training data set of the neural network

Table 3: factors of the main category machine configuration included in the training data set

Sub-category	Factor
Installation	Region
Hardware configuration	Base type (AX3/AX5)
	# of CPR's / SPR's
	Prodrive / Advantech PC's
Software configuration options	SVS present (y/n)
	Speedpack 1/2
	Application SW version

5.3.2 Operation Characteristics

Table 4: factors included in the user profile for the operation characteristics

Sub-category	Factor
# of CO's (per week/month)	# programs run / week
	# of different programs run / week
(Environmental) temperature	Average XY Temperature
	Average PH Temperature

In section 4.2.1 the problem of determining the right time period in which data is collected is described. For the factors **number of programs run / week** and **# of different programs run / week** this problem applies. When comparing different weeks it can be seen that the amount of programs that are run can vary strongly among weeks. To include this factor for most of the machines a one month time period has to be taken into account. This causes to calculate an average number of programs run each week. Averaging this factor makes it harder for the neural network to determine the relationship between the MTBF and the number of (different) programs run / week. The small sample size that is available for the training of the network causes to select only a small number of factors to include in the neural network. Therefore it was decided to exclude the number of (different) programs run / week.

The value of the factor **temperature** is determined by averaging the temperatures of the placement head (ph) and motors that are used for positioning the placement head in the X and Y direction (XY). For these two factors the same problem of determining the right time period in which data is collected, occurs as with the number of (different) programs run / week. Therefore it also was decided to exclude the two different temperature factors from the training set.

The time period in which data is collected causes the deletion of the four factors that were remaining in the main category operation characteristics. This results in the exclusion of this main category in the training set. The issue of the time period in which data is and should be collected is an interesting subject for future research. Related to this issue is whether or not the MTBF should be used as reliability measure. The use of the MTBF causes to select a

large time period in which data should be collected as described in section 4.2.1. A possibility would be to count the amount of failures occurring in a specific period of time, instead of calculating the mean time between failures. This issue is further addressed in section 8.2.4.

5.3.3 Product characteristics

Table 5: factors included in the user profile for the product characteristics

Sub-category	Factor
Board	Dimensions length width thickness
	# of circuits
	# of PA's
	# of packages
	# of components / board
	# of fiducials / board
	# of badmarks
	% CA align placements
Transport	# of indices
	Local / DSF mode
	Zero index (y/n)
	Calculated cycle time
Feeders	# of feeder lanes used
	Type of feeders used (range)
	# of TTF # of ITF <= 16mm # of ITF > 16mm
	Tray feeder
Toolbits	# of toolbit actions
	% CPL placements

Table 5 shows that there are several characteristics of the boards, that are produced, that could be used in the data set. The product placement (PP) program gives a lot of detailed information about what **components** are place on the boards and which **feeders** and **toolbits** are used for the placement of these components. The **dimensions** of the board and the movement thru the machine are also available from the PP file.
As mentioned in the section about operation characteristics there are several different program runs each week. These programs can have little differences in board design such as a few different components. These programs can also contain entirely different board designs, for instance one PP file can be used for the production of memory modules while the other PP file is used for the production of motherboards for PC's. These large differences in board design make it impossible to get one board design that represents all the different board designs that are run in a period of time. This problem causes that for now all the factors of the main category board characteristics should not be taken into account, but a solution should be found to incorporate board characteristics into the data set. This solution should be found

because from experiences within Assembléon it is known that some board designs can cause more machine failures than others.

5.3.4 Machine performance

Table 6: factors that represent the machine performance in the user profile

Sub-category	Factor
Pick&Align performance	PPM_AutoRepickAfterPickError
	PPM_AutoRepickAfterAlignError
	PPM_ManualRecoverAfterPickError
	PPM_ManualRecoverAfterAlignError
Efficiency	% productive time (wrt total time)
	% down time (wrt total time)
	% standby time (wrt total time)
	average recovery time assists [s]
	# pick attempts
	total time [s]

The factors that are discussed until now, except from the temperatures, are characteristics of the machine that can be influenced/changed by operators. The factors included in the main category machine performance (Table 6) give an indication of the pick and align performance of a machine. The machine performance also could help to classify different failures. High numbers of the pick and align performance give an indication that probably a lot of failures are caused during pick or align actions. A pick action is the process where the component, that has to be placed, is picked up from its corresponding feeder. An align actions is the process where the placement robot determines the exact location on the board where the component has to be placed. When a pick or align failure occurs, the machine first automatically performs a repick, when this is not successful a manual recovery has to be performed. The pick and align errors are counted in relation to the total components that are placed. Per million components (PPM) placed the amount of pick and align errors are counted.

The efficiency indicates what percentage of the time the machine was in a certain state. The machine can be in three different time states. The first is **productive time**; this is the time when production has been started, and the AX is busy transporting PCBs, or placing components. The second time state is **down time**, this is the time a machine is down because Maintenance or Service is performed. The third time state is the **standby time**, this is the time in which program is running, and production has been stopped by the operator or the machine is waiting for PCBs to arrive at the runin or leave the runout.

The **average recovery time assist** gives in second the average time it takes for an operator to solve a problem on a machine. This time does not give an indication of the performance of the machine, because the average recovery time is influenced by two factors. The first is the experience of the operator. A more experienced operator is probably capable of solving a failure in a shorter period of time then a less experienced operator. The second factor is the amount of machines an operator has to operate. From experiences within Assembléon it is known that in Asia, where manual labor is cheaper than in for instance Europe, there is often one operator for each machine while in Europe there is one operator that operates several machines. When failures occur at several machines at the same time, in Europe the failures

have to be solved in series while in Asia they can be solved parallel. This could reduce the average recovery time. Because the average recovery time is influenced by these two factors, it first should be investigated how large these influences are and than the average recovery time assist could be incorporated into the data set.

The last two factors are the **number of pick attempts** and **total time** in second. The **# of pick attempts** should be included in the data set because a larger number of pick attempts could increase the chance of a failure to occur. The **total time** is the time the machine is turned on. The total time is used to calculate the mean time between failures for the specific machine, which will be used as output for the neural network as will be described in section 5.3.6. Since the total time is used for calculating the output it is not included as factor. Table 7 summarizes the factors of the main category machine performance that are included in the training data set.

Table 7: factors of the main category machine performance included in the training data set

Sub-category	Factor
Pick&Align performance	PPM_AutoRepickAfterPickError
	PPM_AutoRepickAfterAlignError
	PPM_ManualRecoverAfterPickError
	PPM_ManualRecoverAfterAlignError
Efficiency	% productive time (wrt total time)
	% down time (wrt total time)
	% standby time (wrt total time)
	# pick attempts

5.3.5 Factors included in the neural network

The factors described in Table 8 are included into the data set that will be used to train and validate the neural network.

Table 8: factors included in the neural network

Sub-category	Factor
Hardware configuration detailed	Region
	Base type (AX3/AX5)
	# of CPR's / SPR's
	Prodrive / Advantech PC's
	SVS present (y/n)
	Speedpack 1/2
	Application SW version
Pick & Align performance	PPM_AutoRepickAfterPickError
	PPM_AutoRepickAfterAlignError
	PPM_ManualRecoverAfterPickError
	PPM_ManualRecoverAfterAlignError
Efficiency	% productive time (wrt total time)
	% down time (wrt total time)
	% standby time (wrt total time)
	# pick attempts

The factors described in Table 8 give an answer to the first research question of this master thesis.

Which operational conditions should be included in the model to give a good representation of the real situation?

As mentioned in section 2.3 the research question can be answered by first determining which factors should be included in the neural network to get a good representation of the real situation. These factors were determined during the brainstorm session and the survey. But eventually the practical limitations of collecting data for these factors should also be taken into account. Together with this practical limitation, the factors mentioned in Table 8 form the operational conditions that should be included in the model. These factors give probably not a really good representation of the real situation. Appendix B shows the factors in combination with the relative weights that were calculated from the results of the questionnaire. These relative weights give an indication of how strong a factor is expected to influence the reliability of an Assembléon machine. Of the 16 factors described in Table 8 the factor # of CPR's / SPR's is the only factor that is present in the top ten of factors that are ranked according their relative weight. The application software version is ranked at number 13 while the other factors are ranked in the twenties and forties. These low rankings indicate that these factors are no optimal predictors of the reliability of Assembléon placement machines. Even though the factors are probably no optimal predictors, for now these are the factors of which (sufficient) data can be collected. In section 8.2 recommendations are given how more data can be collected, and how representative data can be collected for factors that are not yet included in this data set.

To create a data set that can be used to train and validate the neural network, the user file is adjusted so it only contains information about the factors that are described in Table 8. The data as it is registered in the user file can not directly be used as input for the neural network. Some of the data have to be coded first so the neural network can incorporate the data. For instance the factor region can not be included in the NN as Europe, Asia, or South America. For each of the values of the factor region, a separate coding is created because the neural network is not capable of using ordinal data. Table 9 shows the coding applied to the factors which don't contain interval or ratio data.

Table 9: coding used for factors in the data set for the NN

Factor	Value	Code
Region	Europe	0 = no 1 = yes
	Asia	0 = no 1 = yes
	South America	0 = no 1 = yes
Base type (AX3/AX5)		0 = AX3 1 = AX 5
Prodrive / Advantech PC's		0 = Prodrive PC 1 = Advantech PC
SVS present (y/n)		0 = no 1 = yes
Speedpack 1/2	none	0 = no 1 = yes
	speedpack 1	0 = no 1 = yes
	speedpack 2	0 = no 1 = yes
Application SW version	milestone version 3.10	0 = no 1 = yes
	milestone version 3.20	0 = no 1 = yes
	milestone version 3.30	0 = no 1 = yes

The data of the factors described in Table 8 are used as input for the training and validation of the neural network. Besides input values, the neural network also needs output values for the training of the network. The next section will described which data is used as output for the neural network.

5.3.6 Output of the neural network

A neural network is capable of determining the relation between input and output as described in section 3.3. The input for the neural network is described in the previous section. To determine what should be used as output of the neural network it is important to refer to the problem description as it is given in section 5.1.

Predicting the reliability performance of an Assembléon machine, by using values of operational conditions, machine configuration, machine performance, and product characteristic.

Mean time between failures

The neural network will be used to determine the reliability of the system. As described in section 2.2 and 2.3 the Mean Time Between Failures is used within Assembléon as a reliability measure. The MTBF will therefore be used as output for the neural network. To complete the data set that is used to train and validate the neural network the MTBF has to be determined for the 43 machines that are included in the data set as described in section 4.2.1. For each of the 43 machines the TBF file is used to determine the amount of failures that occurred during the time period information was gathered for the different input factors. Not all failures registered by the TBF file can be accounted for as failure, therefore the data is corrected.

Failures excluded from the MTBF

The RAM data registers various times, like the total time the machine is turned on, the time production is run, and also the time the machine is down for repairs or maintenance. Failures that occur when zero production hours are registered are not counted as failures because these failures occur while adjustment/repairs are made to the machine. When fixing a failure sometimes trail and error is needed to solve the problem because the exact root cause of the failure is not known. This can result in several failures occurring for each time the machine is restarted. These failures are not related to the reliability of the machine.
The TBF data registers shutdowns of machine, if a machine is shutdown at the end of the day or because production is stopped this should not be counted as a failure. Therefore shutdowns that are registered as "normal shutdown (GUI)" are not registered as failure. The last group of failures that are excluded from being a "real" failure are the power interrupts. Dips in the power supply can cause the machine to shutdown. These power dips are due to lack in reliability of the power supply and not lack in the reliability of the machine.

Productive time

The total time (in second) each machine is turned on, during the time period of interest, can be retrieved from the RAM data. The RAM data also provides the percentage of the total time that each machine is in productive state. Kerstens [Ker07] concluded that to determine the MTBF it is wise to take the productive time instead of total time because failures are most likely to occur during production. If the total time is used this would give the impression the machine has a higher reliability than it actually has. The TBF data registers in which time state the machine is when the machine is shutdown. When looking through the TBF data, it was remarkable that most of the time the machine registered downtime when failures

occurred and not productive time. This contradicted the assumption of Kerstens that most of the failures occurred during the productive time. After discussing this with Assembléon employees it became clear that when for instance a machine has a pick error it first performs a repick and if this is not successful the machine automatically goes to downtime and alarms an operator. The operator can decide to restart the machine (failure is registered) when the error can not be solved. Many errors cause the machine to automatically go to the state downtime before a failure is registered. So many errors are logged as if it occurred during downtime. This is because the machine switches from productive time to downtime just before the failure is registered. Because this switch in time state occurs just before a failure, the failures could be seen as if it occurred during productive time. Therefore the productive time is used in this research to calculate the MTBF as Kerstens suggested.

Calculation of the MTBF
The total time in seconds for the different machines is converted to hours. These hours are multiplied by the percentage of productive time, registered by the RAM data, to acquire the productive time in hours for each machine. The productive time is then divided by the total amount of failures counted in the specific time period. This results in the MTBF for the specific time period for a specific machine.
Nine machines did not register any failures during the time period of interest. For these machines an MTBF could not be calculated. For these machines a different MTBF number is used. Assembléon uses the TBF data of different periods to generate a database in which the reliability performance of different machines is registered. This database contains also MTBF data for the machine of which no failures were registered in specific time of interest. The MTBF data that is registered in this database is calculated by using cumulative failures and cumulative productive time. This implies that from the first failure registered in the database for a specific machine the amount of failures and productive hours are added up, each time a new entry is made. The MTBF is calculated by dividing the cumulative production hours by the cumulative failures. For the nine machines with no failures in the specific time period of interest, this cumulative MTBF is used.
The cumulative MTBF is not used for the other machines because it does not give a representative MTBF for a specific short time period. When for instance a lot of failures occurred just before the time period of interest and adjustments are made to the machine which increased the reliability the MTBF would suggest a poorer reliability. This could also happen the other way around, when a machine had a high reliability (high MTBF) before the time of interest and a lot of failures occur during this period the reliability would still be higher than when the MTBF was calculated for that specific time period.

After the output data set was created the neural network could be created. The next section will describe the procedure of creating the neural network.

5.4 Creating the neural network

After constructing an input and output data set, that can be used for training and validating the neural network, the neural network could be created. The neural network is constructed in Matlab version R2007b. Matlab supports several different possibilities of constructing a neural network. There are two different toolboxes developed for Matlab that can be used. Another option is programming the neural network by using the programming language used in Matlab. The two toolboxes differ in difficulty and in the possibility to adjust specific features of the neural network. Before specifying which construction method was used first the data set that is available has to be discussed shortly because this influenced the selection procedure. As mentioned in section 3.3.4 one of the disadvantages of using a neural network

is the large data set needed to train the network. In section 5.2 an indication is given that the data set is too small for the amount of factors present in the user profile therefore several factors are deleted from the user profile to form the data set. With 16 factors left there is probably still not sufficient data to construct a well working neural network. Therefore two neural networks are constructed. One network will model the relation between machine configuration (8 factors) and the MTBF while the second network will model the relation between the machine performance/efficiency (also 8 factors) and the MTBF. Splitting up the factors and creating two different neural networks is possible because the assumption is made that the factors included in the neural network created for the machine configuration, are independent of the factors included in the neural network created for the machine performance.
The user file is the collection of the different user profiles, as described in section 4.1. In the training set, the term user profile is no longer used because not all factors included in the user profile are used in the training set. For the training set, the term data point is used for the set of values that is collected for the factors of a specific machine.
Section 4.2.1 describes the amount of data that is used for constructing the user file, 43 user profiles are available but for 3 machines no machine configuration could be determined. This results in 40 data points for the data set used for the training of neural network of the machine configuration. For the neural network of the machine performance all 43 data points could be used.
Even though two neural networks (each 8 factors) are created the amount of data points is probably still insufficient to get a good result. When applying the rule of thumb of 10 data points per weighted connection [Zha98] to roughly calculating the amount of data points necessary. This would result, in the situation of 8 input neurons all connected to 8 neurons in the hidden layer (8*8) which each are connected to one output neuron (8), into 72 (64+8) connections which means 720 data points.

In spite of the low amount of data points, but to get a first impression of the performance of the neural network both neural networks are created. Because the choice of training algorithm and transfer function, as described in section 3.3, will not compensate for the small amount of data the most basic toolbox in Matlab is chosen. This toolbox creates a basic neural network in several steps. These steps are:

- Defining the input and output files
- Determine the distribution of the data points over training, test and validation set
- Define the amount of neurons in the hidden layer
- Training the neural network

As just described first step in creating the neural network in the Matlab toolbox is defining the input and output files. The second step is determining the distribution of data over training, test and validation set. This distribution is given in percentages of the total data set that is available. The training set is used to train the data so the weights between neurons can be adjusted. The test data set is used to determine when the training should be stopped, this to prevent over training. The training is automatically stopped when generalization stops improving this is indicated by an increase in the mean square error of the test samples. When a network is over trained the network does not predict the outcome, but memorizes the relation of the training set it is presented with. The validation set is a data set that is not used during training this data set is new to the network and is used to determine the predictive performance of the trained network.
In the next step the amount of hidden neurons can be chosen. The size of the hidden layer can influence the predictive performance of the neural network. There are no specific guidelines for determining the size of the hidden layer, this is a process of trial and error. Therefore a

structured experiment is setup, in which the amount of hidden neurons is changed. This experiment can help to determine the influence of the amount of hidden neurons in relation to the predictive performance for each specific network.
The toolbox generates a two-layer feedforward network trained with a back propagating learning rule (Levenberg-Marquardt) [Dem07]. The transfer functions that are applied by the toolbox are Tan-sigmoid [Dem07].

When the network is created the network can be trained. After the training of the network, the Mean Squared Error (MSE) and the regression value (R) are presented for each of the three data sets (training, test, and validation). The Mean Squared Error is the average squared difference between outputs and targets. The output is the MTBF calculated by the neural network using the values of the different factors. The target is the MTBF as it is registered in the output file. Lower values in the MSE indicate a better fit of the model. The regression value measures the correlation between outputs and targets. An R value of 1 means a high correlation, 0 no correlation.

5.5 Performance of the neural network

As described in the previous section the amount of neurons in the hidden layer have to be determined by trial & error. Therefore the size of the hidden layer is varied between 4, 8 and 16 hidden neurons. Also the influence of the division in the data set between the training, test and validation data set are investigated by experiments. Therefore the data set is split up in the first experiment in 60% training, 20% test, and 20% validation, while in the second experiment the data is split in 75% training, 15% test, and 10% validation.

Each of the two networks is trained 4 times in a row to see what the influence of the sample selection is. Each time the network is (re)trained the toolbox randomly divides the data set in the division earlier selected. For each configuration the MSE and regression value (R) are registered to get an indication of the performance of the network.

5.5.1 Machine configuration

Table 10 shows the results of the different neural network configurations of the machine configuration that are described above. The values in the first column indicate how many neurons were used in the hidden layer. The second column shows the division in percentages of the total data set into training, test, and validation set. The numbered columns show the results (MSE, R) of the 4 different training sessions. The last two columns show the average value and standard deviation of the MSE and R value calculated over the four trainings runs.

Table 10: Results of the different neural networks configurations created for the machine configuration

hidden neurons	division data set (%)	data set	1		2		3		4		Average		Standard deviation	
			MSE	R	MSE	R	MSE	R	MSE	R	MSE	R	MSE	R
4	60	training	21162	0,772	19645	0,674	30292	-0,388	62298	0,304	33349	0,341	19864	0,526
	20	test	39476	0,326	46436	-0,187	61447	0,156	48729	-0,138	49022	0,039	9170	0,244
	20	validation	91483	-0,071	18528	0,43	33972	-0,262	52944	-0,566	49232	-0,117	31488	0,418
8	60	training	4687	0,927	36903	0,288	44144	0,297	22076	0,733	26953	0,561	17456	0,320
	20	test	51015	0,281	57344	-0,089	67289	-0,17	122025	0,387	74418	0,102	32437	0,273
	20	validation	88154	-0,529	38410	0,557	22616	0,343	176484	-0,816	81416	-0,111	69259	0,664
16	60	training	7179	0,898	43574	0,440	48001	0,597	4023	0,939	25694	0,719	23308	0,240
	20	test	33550	0,246	20871	0,691	127242	0,610	23601	0,377	51316	0,481	50910	0,206
	20	validation	157332	-0,087	54903	0,073	230358	-0,351	50677	0,003	123318	-0,091	86741	0,186
8	75	training	244518	-0,126	7657	0,849	42988	0,554	7195	0,866	75590	0,536	113860	0,464
	15	test	44933	0,606	62589	0,466	43086	0,598	66183	-0,324	54198	0,337	11879	0,445
	10	validation	74362	0,356	22144	-0,073	29321	-0,657	53432	0,710	44815	0,084	23814	0,589

The results in Table 10 show that the performance indicators (MSE, R value) show a large variation between each training run and between the different configurations. The validation data set is not used during the training of the network. The network is therefore not capable of memorizing the combination between input variables and the MTBF. The results for the validation data set therefore give an indication of the predictive performance of the neural network. The predictive performance of the trained neural network on the validation set varies strong between each of the 4 training runs. Table 10 shows that for each of the four different configurations the standard deviation of the regression value is larger than the average regression value calculated over the 4 training runs. The regression value measures the correlation between output of the neural network and the target value. When the predictive performance of the neural network is high the regression value would be close to one. The highest average regression value for a validation set in Table 10 is 0,084. This low value together with the large variation indicates that the performance of the network is insufficient.
The bad performance of the neural network can have several reasons. The small sample size causes the neural network to be trained in an insufficient manner. Another reason could be that due to the factor reduction that had to be performed, only factors are included in the neural network that are not capable of predicting the MTBF. Probably the combination of both arguments just described cause the bad performance of the neural network.

5.5.2 Machine performance

Table 11 displays the results of the different training runs performed for the different configuration of the neural network created for the machine performance. As with the neural network of the machine configuration, the amount of hidden neurons and the division of the data set over the training, test, and validation set is varied. The results (MSE, R) of each of the 4 training runs are displayed in Table 11. As with the neural network created for the machine configuration, the results show a large variation in performance of the network. This neural network also shows that for each of the four different configurations the standard deviation of the regression value, of the validation set, is larger than the average regression value calculated over the 4 training runs.
This large variation is probably caused by the small data set available and the use of factors in the neural network that are not capable of predicting the MTBF in a sufficient manner.

Table 11: Results of the different neural networks configurations created for the machine performance

hidden neurons	division data set (%)	data set	1		2		3		4		Average		Standard deviation	
			MSE	R	MSE	R	MSE	R	MSE	R	MSE	R	MSE	R
4	60	training	20439	0,636	18925	0,643	18317	0,615	21302	0,562	19746	0,614	1368	0,037
	20	test	21022	0,109	32071	0,314	39299	-0,035	30129	0,669	30630	0,264	7523	0,305
	20	validation	41450	0,058	49898	0,047	37654	0,364	25559	0,016	38640	0,121	10111	0,163
8	60	training	13157	0,673	105229	-0,037	15885	0,634	14939	0,569	37303	0,460	45298	0,334
	20	test	47160	0,591	20024	0,668	28271	0,225	22087	0,731	29386	0,554	12357	0,227
	20	validation	101494	-0,013	47755	0,658	55144	0,236	60186	0,008	66145	0,222	24113	0,312
16	60	training	6889	0,857	25493	0,287	22323	0,622	10315	0,726	16255	0,623	9040	0,244
	20	test	49426	0,491	32027	0,489	62846	-0,262	46112	0,315	47603	0,258	12656	0,357
	20	validation	121570	-0,814	54065	-0,094	36405	0,242	86156	-0,438	74549	-0,276	37507	0,454
8	75	training	23117	0,534	32953	0,466	63051	0,244	27338	0,584	36615	0,457	18079	0,150
	15	test	16513	0,767	20527	0,004	1594	0,899	10615	0,219	12312	0,472	8224	0,429
	10	validation	27699	0,253	5371	0,809	39324	-0,362	29669	0,510	25516	0,303	14358	0,498

5.5.3 Influence of the small data set

The strong variation in regression values between training runs and the low average regression value for the validation set, show that the predictive performance is bad for both neural network described in the previous two sections.
One of the reasons for this bad predictive performance, as mentioned in section 5.5.1, is the small data set. With the small amount of data points available the performance could be strongly influenced by the sample selection.
Due to practical limitations in the data collection and the small sample size, only several factors could be included in the neural network. The selection of factors as described in section 5.3 could also influence the predictive performance of the created neural networks.
The bad performance of the neural network which causes large fluctuations between the different training runs, makes it impossible to determine the influence of the amount of neurons in the hidden layer and the deviation in the data set. Therefore it is hard to answer the second and third research question. Which are:

2. What influence does the amount of (training) data have on the accuracy of the neural network?

3. What influence does the addition of extra input factors have on the accuracy of the neural network?

In order to be able to answer the second and third research question, it is important that the neural network can be trained in a sufficient manner so the performance doesn't fluctuate so strong for each training run. Because of these fluctuations it is not possible to compare the performance of the different configurations in the neural network. Not being able to compare the performance of the different neural network configurations, makes it impossible to investigate the influence of the amount of training data and input factors on the accuracy of the predictions made by the neural network.

The fourth research question can be answer in relation to the amount of data now available and the neural network that can be created with this data. The fourth research question is:

4. Can the neural network be used to compare different software releases under different operational conditions?

With the data that is available now, the neural network can not be used to describe in a short period of time whether a new release performs as reliable as or better than the old and existing release. In order to answer this research question, the most important step would be collecting more data. When more data is collected it can be investigated whether a trained neural network is capable of correctly estimating the MTBF and therefore can be used to determine whether software is ready for release, as described in section 2.3.
To increase the sample size data could be collected for the same machines that are used in this research but for more periods of times. Another possibility would be collecting data of a larger variety of machines. To determine whether more machines should be included in the data set, first the time period it would take to collect sufficient information with the 31 unique machines used in this research should be calculated. The following assumptions are made for this calculation.

- The data is collected over a one month period
- The network contains 8 hidden neurons
- All neurons are fully connected
- The rule of thumb of 10 data points per connection is applied

The 8 input factors in combination with 8 hidden neurons and 1 output neuron are connected by 72 connections, as described in section 5.4. To train the network in a sufficient manner it would require 720 data points. The 31 unique machines would need to gather information for 24 consecutive months. This 2 year time period would be to long, because Assembléon releases one or two software updates each year. Therefore more machines should be included in future research.
The small data set is an important issue when training the neural network. The training will probably not be optimal due to the small sample size, which results in a bad predictive performance of the neural network. Besides not optimally training the network the bad predictive performance of the neural network could also be caused by the selection of the factors. As described in section 5.3 factors are excluded from the neural network for several different reasons. The factors that are included in the neural network could have a small influence on the reliability of the machine, and therefore be probably bad predictors of the MTBF. Including these bad predictors would cause the poor predictive performance of the neural network.

The performance indictors of the neural network that are used in this research are the MSE and regression value. These indicators are used as measure to determine whether the performance of one configuration is better than the other. To determine whether a neural network is capable of comparing different software releases under different operational conditions, it is important to have a benchmark against which the predictive performance of the network can be compared. A simple method for determining a benchmark value, to compare the performance of the neural network with, would be a linear regression model. This method searches for linear relations between several independent (input) variables and a dependent variable (output). The next chapter will demonstrate how a linear regression model can be used to generate a benchmark value.

6. Linear regression analysis

The results of the neural network are strongly influenced by the small sample that is available. The results of a regression analysis are, just like the neural network, influenced by a small sample size. Therefore, this role of the sample size is shortly described before performing a regression analysis on the data set that is used for the creation of the neural network.

6.1 Influence of the sample size

The sample size can influence the statistical test by either making it insensitive (a small sample size) or overly sensitive (a large sample size). A small sample size (less than 30 observations) is appropriate for analyzing a simple regression with one independent variable. In this situation only strong relationships can be detected with a degree of certainty. When the sample size is large (more than 1000 observations) almost all relationships are found to be statistically significant [Hai05].
As just described, the sample size can have influence on the relations detected by the linear regression analysis. What size should the sample size be for a regression model? As with the neural network where the rule of thumb of 10 data points per weighted connection is used, for a regression model the ratio should never fall below 5:1. This means 5 observations for each independent variable that is used in the analysis. Although the minimum ratio is 5:1 it is desired that for each independent variable there are 15 to 20 observations. [Hai05]

The data sets that are used for constructing the two linear regression models (machine configuration, machine performance) in this research contains 8 independent variables and has 40 observations for the machine configuration and 43 observations for the machine performance. This results in the minimum ratio of 5:1. This low ratio indicates that only very strong relationships between the independent variables and MTBF will be found to be statistically significant.

6.2 Regression model

To perform the linear regression analysis, SPSS 16.0 is used. First the data set that is also used for training the neural network is imported into SPSS. Because nominal data can not directly be incorporated into a regression model, SPSS transforms variables that are labeled as nominal value automatically into dummy variables. For the machine configuration, the amount of compact and standard placement robots and the MTBF were the only variables not labeled as nominal variables. The variables included in the machine performance contain no nominal values. Appendix D shows the coded values for the nominal variables used for the machine configuration. First the regression model of the machine configuration will be discussed after which the regression model of the machine performance will be described

6.2.1 Machine configuration

To compare the predictive performance of the neural network with the regression model all 8 input variables of the neural network were selected as independent variables where the MTBF was used as dependent variable. Appendix E shows the results of the regression analysis performed with the machine configuration as dependent and the MTBF as independent variable. The regression model has an adjusted R^2 of 0,209, which means that only 20,9 % of the total variance can be explained by the regression model. The F ratio of the model is 2,29 which indicates considering the sample that is used, that the model can explain 2,29 times more variation than using the average. Both these numbers indicate that the model

is not capable of describing accurately the relationship between the independent (machine configuration) and dependent variable (MTBF).

6.2.2 Machine performance

The results of the regression analysis of the machine performance are presented in appendix E. The results show that only 4% of the total variance can be explained by the model (adjusted R^2 = 0,044). The F ratio of 1,23, indicates that the model can explain 1,23 times more variation than using the average, but only with 68,9% certainty (sig. 0,311). These results indicate that the independent variables (machine performance), as they are used in this model, are bad predictors of the dependent variable (MTBF)

6.3 Regression model as benchmark

As indicated in section 6.1 the sample size is too small to generate a significant regression model for the amount of parameters included in the model, as described in the previous two sections. As with the neural network, one of the first steps to increasing the performance of the regression model is collecting more data. When using the advised ratio of 20 data points for each independent variable, as described in section 6.1, it would mean 160 data points have to be gathered for a regression model with 8 independent variables. To train the neural network in a sufficient manner, 720 data points are required. So when enough data is collected for the development of a neural network, this will also be sufficient for performing a linear regression.

The regression model should be created as a benchmark to compare the predictive performance of the neural network with. The performance of the neural network is described by the mean square error and the regression value of the validation data set. This data set is not used earlier for training or test purposes, so the network has no possibilities of memorizing the relations between input and output. For the regression model also a validation data set should be created. This validation set can be used to determine the predictive performance of the regression model. As performance indicator of the different models, the mean square error can be used. In a future research it would be interesting to investigate whether the MSE is the most appropriate indicator to use for the comparison of the performance of the neural network and the regression model. When the predictive performance of the neural network is poorer than the regression model the configuration of the neural network should be altered or the conclusion should be drawn that a neural network is not an ideal tool to model the relation between input and output.

7. Conclusions

This chapter will summarize the conclusions of this research. At the beginning of this project several research questions are defined that are used as guide to perform this research. Conclusions can be drawn from the answer given to these research questions.

The first research question was:

1. Which operational conditions (factors) should be included in the model to give a good representation of the real situation?

The following factors should be used to model the reliability of an Assembléon.

- In which region the machine is placed
- Whether the base type is an AX3 or AX5
- The number of compact placement robots present in the machine
- The number of standard placement robots present in the machine
- Whether the placement controller is manufactured by Prodrive or Advantech
- The presence of Setup Verification System (SVS)
- Which speedpack is used
- Which software version is running on the machine
- PPM numbers for the AutoRepickAfterPickError
- PPM numbers for the AutoRepickAfterAlignError
- PPM numbers for the ManualRecoverAfterPickError
- PPM numbers for the ManualRecoverAfterAlignError
- The percentage of time the machine is in productive state
- The percentage of time the machine is in downtime state
- The percentage of time the machine is in standby state
- The number of pick attempts performed by the machine

The factors that are selected for modeling the reliability of an Assembléon machine are in this research mainly selected on practical grounds, whether or not data could be collected. Because the factors are mainly selected on practical grounds, factors are excluded that could have a large influence on the reliability of an Assembléon machine and thereby might be good predictors. This exclusion of the factors may cause the model to suffer in predictive performance and therefore lack in representing the real situation.

The second and third research question, are focused on the relation between the design of a neural network and its performance. The two research questions are:

2. What influence does the amount of (training) data have on the accuracy of the neural network?

3. What influence does the addition of extra input factors have on the accuracy of the neural network?

To answer the second and third research question, it is important that the neural network can be trained in a sufficient manner so the performance doesn't fluctuate strong for each training run as seen in section 5.5. Because training of the network can not be performed in a sufficient manner, due to the lack of large amounts of data, it is not possible to compare the performance of the different configurations in the neural network. Not being able to compare

the performance of the different neural network configurations, makes it impossible to investigate the influence of the amount of training data and input factors on the accuracy of the predictions made by the neural network. Therefore the second and third research question could not be answered.

The fourth research question can only be answer in relation to the amount of data that was available during this research. The fourth research question is.

4. Can the neural network be used to compare different software releases under different operational conditions?

With the small data set now available, the neural network can at this moment not yet be used to describe in a short period of time whether a new release performs as reliable as or better than the old and existing release.

The small data set has strongly influenced the capability of answering the research questions. Therefore collecting more data is an important step in a future research. It would take 2 year to gather enough information to create a neural network, with 8 input factors and 8 hidden neurons, with the 31 machines used for this research. Therefore more machines have to be used for data collection.

The availability of only a small sample size was largely caused by the fact that data collection and analysis was a manual and therefore time consuming activity. Automatic data collection and analysis would easily increase the amount of data points.

The use of the mean time between failure (MTBF) as reliability measure, introduces some problems with the data collection. To use the MTBF requires the time period in which data is collected to be sufficient long (e.g. one month) in order to register a failure. This long time period which is required for the MTBF causes problems with the collection of data for other factors. Factors like change in placement program could better be measured in a one week period instead of a month. When data for these factors is collected for longer periods of time the predictive influence of these factors on the reliability could decrease. This decrease in predictive influence is the result of calculating an average value for a factor, like the number of placement programs run, over a time period. When this time period is too long, large increases in the value of a factor could not be noticed. These large increases could result in a failure of the machine. So when these large increases are not noticed it is also not possible to find the relation between these increases and the failures that could be related to these increases in the value of a factor. This could influence the predictive influence of a factor, as described in section 4.2.1.

8. Discussion

As a solution to a bad performance of the neural network and the regression model, the collection of more data is proposed as first solution. Besides collecting more information there are two other possibilities, namely decreasing the amount of factors or using another method for predicting the reliability. In this chapter first the reduction of the amount of factors will be discussed, after which the collection of more data will be discussed more elaborate. As third the use of time series analysis as a different prediction method will be discussed. All three possibilities will be discussed in relation to the data and factors available within Assembléon.

8.1 Reducing the amount of factors

The reduction of the amount of factors included in the neural network as well in the regression model make it possible to use a smaller sample size. Section 5.3 describes de reduction of the amount of factors that are included in the training set of the neural network. This reduction is necessary because a small sample size is available. The reduction of the amount of factors included in the model probably already had a strong negative influence on the predictive performance of the neural network. Reducing the amount of factors even further would probably not result in a better model.
When choosing to reduce the amount of factors it is of importance that only the factors are excluded that have a small influence on the predictive performance of the model. This could be because there is no or a weak relationship between the factor and the output or because two or more factors are strongly related to each other. This strong relation between factors makes it possible to only use one of the factors because this factor can represents the other factors. To determine which factors to include and which to exclude, a factor analysis can be performed. The next section will describe the procedures related to factor analysis.

8.1.1 Factor analysis

Factor analysis is a technique which analysis the structure of the interrelationships (correlation) among a number of variables. This makes it possible to summarize information contained in several different variables into a smaller set of new variables with a minimum loss of information [Hai05]. The new variables are called factors.

As with the other two methods (NN and regression model) described earlier in this thesis, sample size plays an important role in performing a factor analysis. Hair [Hai05] describes that generally researchers do not factor analyze a sample that is smaller than 50 observations, but preferably the sample size should be 100 observations or larger. As with the regression analysis the minimum ratio between the amount of variables and observations is 1:5, but preferably the ratio should be between the 1:10 and 1:20. With a high observation-per-variable ratio the chances of overfitting the data is minimized. The sample size of the two data sets (machine configuration and machine performance) are below 50 observations, but the ratio of 5 observations per variable can be met. This small sample size imply that the results found in the factor analysis should be interpreted with caution because interrelationships that are found may in reality be not as strong as suggested and other interrelationships may not be found at all.

The factor analysis is performed in SPSS 16.0. For both the machine configuration and machine performance data sets the same procedure is used.

- Determine the amount of factors that have to be created
- Creating a factor matrix
- Applying rotation to the factor matrix
- Selecting a variable to represent the newly created factor

As first step in the factor analysis the amount of factors that should be created is determined. The latent root criterion is selected as method to determine the amount of factors, because this method is most commonly used. Hair [Hai05] describes in chapter 3 of his book more elaborate what the latent root criterion is and how it is applied. Hair also mentions that this method works most optimal when 20 to 50 variables are used. In this research only 8 variables are used which could result in the selection of too few factors. With the latent root criterion it is determined that for the machine configuration four factors should be created, while for the machine performance three factors should be created.
After the amount of factors is determined, the factor matrix is created in which the factor loadings are displayed for each variable. The factor loadings are the correlation between each variable and the factor. Table 12 shows for instance that the variable region has a factor loading of 0,426 on the first factor, 0,235 on the second factor and so on. A high factor loading makes the variable representative for the factor.
To achieve a simpler and theoretically more meaningful factor matrix, rotation can be applied to the original factor matrix. By applying rotation to the original factor matrix, redistribution of variance over the factors will take place. The process of rotation is described more elaborate in Hair [Hai05]. Eventually the rotated factor matrix is used to determine which variable could best represent (explain the largest portion of variance) each different factor. The variable with the highest significant factor loading is selected to represent the factor. Hair describes that the significance of a factor loading depends on the sample size that is available. The sample size of 40 data points requires the factor loading to be above 0,75 to be statistically significant. Appendix F summarizes the results of the factor analysis performed for the machine configuration and machine performance.

Machine configuration

Table 12 shows the rotated matrix of the machine configuration.

Table 12: rotated factor matrix for machine configuration

	Factor			
Variable	1	2	3	4
Region	0,426	0,235	-0,539	0,146
Base type	0,515	0,569	0,478	-0,245
CPR	0,979	-0,021	-0,111	0,007
SPR	-0,835	0,254	0,364	-0,079
Controller	-0,011	0,867	-0,232	-0,179
SVS	-0,151	0,825	0,006	0,140
Speedpack	-0,178	-0,061	0,838	0,139
Software version	0,054	-0,030	0,044	0,964

Extraction Method: Principal Component Analysis.
Rotation Method: Varimax with Kaiser Normalization.

From the table above it can be concluded that the first factor can best be represented by the variable amount of compact placement robots (factor loading 0,979). The second factor is represented by the variable type of placement controller (0,867). With a factor loading of 0,838 the variable speedpack represents factor 3. The final factor is represented by the variable software version (0,964).

The four variables selected via factor analysis of the machine configuration are used in a linear regression model to investigate whether the predictive performance of this model is better than performance of the regression model developed in section 6.2.1 which contained all 8 variables. The results of the regression analysis of the machine configuration are presented in appendix G. The results show that 23,3% of the total variance can be explained by the model (adjusted R^2 = 0,233). The F ratio of 3,957, indicates that the model can explain 3,957 times more variation than using the average. The regression model of the machine configuration described in section 6.2.1, has an adjusted R^2 of 0,209, and an F ratio of 2,29. The regression model with only 4 variables, selected via factor analysis, shows a better predictive performance (higher R^2 and the F ratio) than the model with 8 variables. A part of this increase in predictive performance is caused by the fact that the data set of 40 data points is used in the second model to model the relation between the dependent variable and only 4 independent variables. Whereas the first model used 40 data points to determine the relation between 8 independent variables and a dependent variable. The first model has an observation-per-variable ratio of 5:1 where the second model has a ratio of 10:1. This higher observation-per-variable ratio makes it possible to find a stronger relationship between independent and dependent variables.

Machine performance

Table 13 shows the rotated matrix of the machine performance

Table 13: rotated factor matrix for machine performance

	Factor		
Variable	1	2	3
Autorepickafterpickerror	,963	-,023	,033
Autorepickafteralignerror	,979	-,021	,033
Manualrecoveryafterpickerror	,967	-,020	-,095
Manualrecoveryafteralignerror	,962	-,036	-,054
Productive time	-,032	,999	,015
Down time	-,025	,999	,003
Standby time	-,023	,999	-,020
Pick attempts	-,038	-,002	,998

Extraction Method: Principal Component Analysis.
Rotation Method: Varimax with Kaiser Normalization.

From the rotated factor matrix shown above three variables are selected to represent each of the three different factors. The four variables representing the pick and align errors all have a high factor loading on factor 1. The three different time states all have a high factor loading on factor 2. When more than one variable has a high factor loading on a specific factor, not always the variable with the highest loading is selected to represent the factor. In this situation also the loading on the other factors are taken into account. A low factor loading on the other factors indicates that the variable is not strongly correlated with the other factors. The variable Autorepickafteralignerror has the highest factor loading on factor 1 and the lowest factor loading on the other two factors. The variables that represent the three different time states all have a factor loading of 0,999 on the second factor. The variable downtime is selected to represent factor 2, it has the lowest factor loading factor 3 and last but one on the first factor. The variable pick attempts is selected to represent factor 3.

The three variables selected via factor analysis of the machine performance are used in a linear regression model. The results of the regression analysis are shown in appendix G. The results show that even though the regression model with 8 variables, as described in section 6.2.2, has a bad predictive performance (adjusted R^2 = 0,044, F ratio = 1,23). The regression model with 3 variables has an even worst predictive performance. The model has an adjusted R^2 of -,071 and can explain only 0,09 times more variation (F ratio = 0,09) than using the average, but only with 4,5% certainty (sig. 0,965). Selecting only 3 variables from the eight variables that first represented the machine performance decreased the predictive performance of the regression model. This bad predictive performance can be caused by the selection of 3 variables that are not (linear) related to the independent variable MTBF.

The reduction of the amount of factors included in the regression model make it possible to achieve a higher observation-per-variable ratio which could result in a better regression model. This probably happened with the regression model of the machine configuration. Reducing the amount of factors has on the other hand the disadvantage that the model will probably lose some predictive performance. This probably happened with the regression model of the machine performance, because variables are excluded that in some extent have a linear relation with the independent variable (MTBF). The next section describes how the sample size could be increased.

8.2 Increasing the sample size

The small sample size that is available until now causes a lot of techniques not to perform in an optimal manner. Increasing the sample size is therefore an important step in successive research. Section 4.2 describes how data is gathered for the user profile. Before large amounts of data can be gathered to increase the sample size, first the problems have to be solved encountered during the data collection. Some of the most important problems are:

1. Which data should be included in the user profile; when lots of information is available it is important to determine which data is interesting to collect in a user profile. But because data is collected from different sources this sometimes results in files that can not be used or are incomplete.
2. The data has to be gathered from different sources. Each of these sources has a different way of registering time. RAM data files for example give a time period in which data is collected while TBF data is only registered on specific date and time when a startup or shutdown occurs. Program files only indicate when they are created and not when they are run. This makes it hard to combine all different files to get correct combination of performance for a specific period or instance of time.
3. The data transformation and analysis is at the moment performed manually which makes this process very time consuming and introduces errors. Collecting, and manually transforming and analyzing the large sample size needed for the training of the neural network would take a lot of time. The transformation and analysis is time consuming because different file types collected from the machine have to be converted into Microsoft Excel files. After the excel sheets are created, data is spread throughout the document on different sheets. Also pivot tables have to be created for collecting data. All these processes take time and can introduce errors when not performed correctly.

4. Collecting all data needed for the large sample size is hard from practical point of view, because data collection is not a priority of the operators controlling the machine. In order to collect all the needed data from a machine it is necessary to get the different files from the machine itself. This means that this is an action that an operator has to perform. Collecting this data is often not a priority of the operator because the focus is on the production so that as much boards as possible are produced. This makes the collection of large amounts of data harder. For the collection of RAM data there is also another obstacle, the machine has to be stopped in order to collect the data. This causes that the data can not be gathered at any given time but only when the machine is not producing boards.

8.2.1 Which factors to include in the user profile

The first problem, the large amounts of factors that are available and which to incorporate in the user profile, is discussed in chapter 4. In chapter 4 the selection of factors is performed by first determining how significant the influence is of a factor on the reliability of the system, according to Assembléon employees. Than the factors that are identified as influential factors, are investigated from a practical point of view. For some of the influential factors no representative data could be collected. For instance the factor user skill, which indicates how well an operator knows how to work with an Assembléon machine, could not be quantified. In a follow up research it would be interesting to investigate how these influential factors which are not practical in use, could be incorporated in the user profile.
A possibility could be to try to quantify these factors in an indirect manner. For instance the user skill could be related to a certain PPM number or failure recovery time. When a significant relationship is found between the user skill and another factor, that can be quantified, this factor can represent the user skill.
Another possibility could be to gather over a large period of time information of one specific machine. When assuming that this machine is operated by the same group of operators and is stationed at the same location over time, the influence of for instance, the environment and operators on the reliability of the machine could be kept constant. When this machine shows the same fluctuations in reliability over time as several other machines do, it could be that the environment or operator has no large impact on the reliability of the machine. Therefore it is not a problem that these factors cannot be quantified.
In the previous section, it was explained that by reducing the amount of factors the sample size that is necessary to create an accurate prediction model will also decrease. The first problem just described, pleads for the incorporation of more factors to get a more accurate prediction method. Including more factors would also mean gathering larger amounts of data. The only way to gather these large amounts of data would be by creating an automatic data gathering tool. The subject of automatic data collection is described after the three other problems encountered during the data collection are discussed.

8.2.2 Practical problems encountered during data collection

The last three problems described in section 8.2, are practical problems encountered during the data collection. The differences in types of time registration in the different file types make it harder to collect information for a specific period. The differences in types of time registration are present because, for instance in the TBF file only time is registered when an event occurs. It would be illogical to register other times that are not related to failure data.
The conversion of files logged by the machine into excel files is time consuming because there is a lot of information registered by the machine. To be able to acquire specific information from the created excel files it is necessary to create pivot tables that organize information in a practical manner. Other data is scattered throughout the excel file, which

may cause incorrect data collection when data has to be gathered form a large amount of machines.
The fact that information has to be gathered physically from a large amount of machines makes the data collection harder. Assembléon has agreements with specific customers that make remote data collection possible. When more data has to be gathered it would be wise to select machines that have remote data collection capabilities. This enables to periodically collect all the data that needs to be gathered to construct a clear user profile.

8.2.3 Automatic data collection

The automatic data collection could also help with solving the problems of combining different files due to time registration and the problem of time consuming manual data collection. Thereby easily increase the sample size. By creating a software application it would be possible to automatically gather all the information needed for the user profile at once. The software tool could gather data from the different file types and organize it in such a manner that it could directly be incorporated in the user profile. This would make it possible to gather the large amount of data (720 data points) that needs to be gathered to create a reliable user profile and neural network.

Section 5.3 describes the evaluation of factors in the user profile, this to reduce the amount of factors that could be incorporated in the neural network. As mentioned earlier it is important to incorporate those factors in the model, which have a large influence on the reliability of the machine. Another possibility to increase the sample size would be to collect data in a shorter time period, this would make it easier to collect larger amounts of data. A shorter time period will cause problems with the use of the MTBF as reliability measure. The problem of choosing the right time resolution, in which data is collected, will be described in the next section.

8.2.4 Choosing the right time resolution

The user profile as it is constructed in this research poses several problems. One of the problems is the time period in which data is collected. For this research, a one month period was used for a large portion of the machines. This one month period could not be used for all machines because, data was gathered for a specific period before the research started or data was not registered for a long enough period to use a one month time period.
The one month time period used seemed to be the correct time resolution for the machine configuration. The machine configuration doesn't change to often so a one month period can be used to give a representative picture of the machine configuration.
Section 5.3.2 indicated that the factor number of programs run per week could not be incorporated because an average number has to be calculated for the amount of programs run per week. Averaging this value would probably reduce the predictive value in relation to the MTBF. The information that is related to the amount of programs that are run, would be interesting to gather in a time resolution of one week or shorter.
The MTBF is used as output/ dependent factor. The time period that is used for the MTBF should be the same as the time period in which information is gathered to ensure a good representation of the influence of the input/ independent factors in relation to the MTBF. As just described it would be interesting to gather data in different time resolutions. The MTBF should also be gathered in different periods, so it is the same as in which input factors are collected. Collecting MTBF data in a one week period would result in lots of weeks in which no failures occur and no MTBF could be calculated. In the one month period in which data was now collected there were some machines with no failures, this amount would increase when a one week period was chosen.

As described above, collecting information over time poses practical problems. With collecting information over a period of time also valuable information is lost. As mentioned earlier values have to be averaged. This is not possible for all factors. The placement program contains a lot of valuable information but if two different board designs are run it is not possible to determine an average board that represents both designs. Also specific events like a placement program that is run or increasing PPM numbers can result in failures. Failures are often triggered by actions or changes in the system. By looking over a certain period of time these special events would not be noticed.

8.2.5 Recording of events

Failure, just like changes in the system, can be seen as events. If all failures and changes in the system are logged as specific events over a time line, the problem with the need of different time resolutions would not occur. Each failure would be seen as a separate event, just like the change of software version or change in placement program. When registering changes and failures as an event it could maybe be possible to detect specific events just before a failure occurs. Using this method, could for example show that each time a specific placement program is run, a failure occurs. This indicates that there are problems with this specific placement program. This kind of specific information could not be gathered when the amount of placement programs are registered over a time period.
Figure 9 gives an example of how the data could graphically be presented. On the X axis there is a time line registered. Each failure and event is represented by a marker placed on the time of occurrence, the figure is not based on actual data.

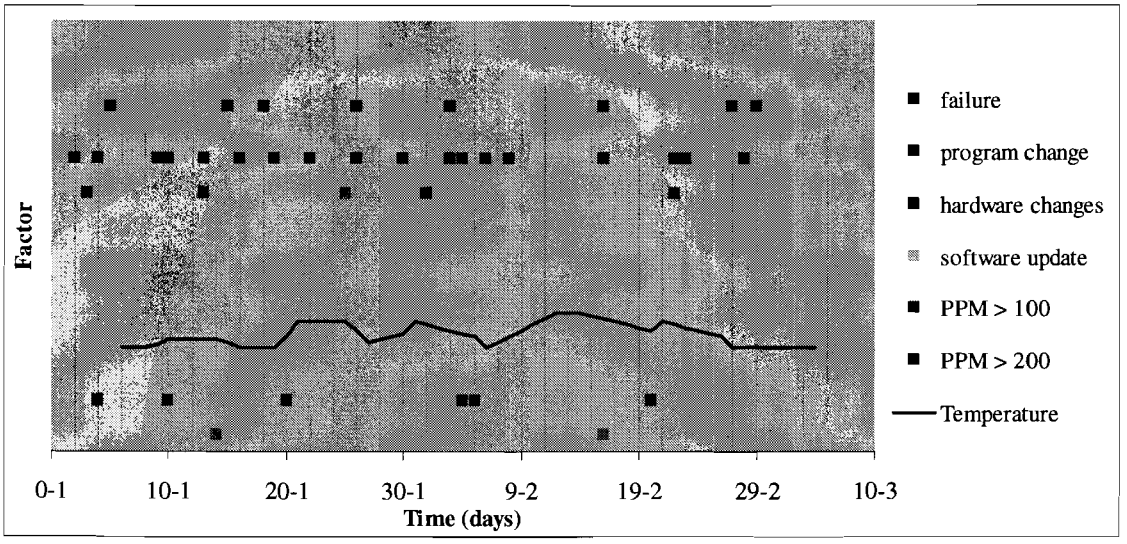


Figure 9: example of changes and failures registered as events over time

Events like change in placement program or a hardware swop can easily be identified and registered as an event. Variables like temperatures or PPM numbers are measured continuously over time. Displaying these variables can be done in two different ways. First the data can be plotted over time, so a continues graph is created. The second possibility is creating categories of value ranges and then registering the moment when a factor changes from one to the other category. If for instance the PPM number for alignment errors is normally around 80, categories could be created with steps of 100, so category 1 would be a PPM number from 0 to 100, category 2 a PPM number from 101 to 200 and so on. Each time the PPM reaches for instance 130 an event is registered in the second category.

8.3 The use time series analysis

The focus of this research lies on creating a neural network that can predict the reliability of an Assembléon machine. Due to lack in sufficient training data the network probably does not perform as intended. Besides the small data set it could also be possible that a neural network is not the optimal technique to use to predict the reliability of an Assembléon machine. Therefore the possibility of using a different model should also be considered. In the previous sections the use of the MTBF as reliability measure is discussed. This section also describes the possibilities of registering events over time. Time series analysis is a technique that can analyze data that is registered over time.

Sequentially registered data over time are called time series. Time series are collected in a variety of fields, like economics, marketing, demographic, and process control [Cha04]. When observations are made continuously through time the time series is said to be continuous. Time series of PPM levels and the temperature are continuous time series. The variables that can be registered as an event in time like a change in placement program are called point processes. Describing which techniques could best be used in the situation where continuous data and point processes data are available is beyond the scope of this research. Investigating the use of time series for reliability purposes at Assembléon could be interesting for future research.

Analyzing time series can have different objectives [Cha04]. By plotting the observations against time a time plot is created. This can help with identifying seasonal patterns and possible outliers in the data set. When observations are taken on two or more variables, it may be possible to use the variation in one time series to explain the variance in another time series. This may help to explain and understand the relation between variables. Time series can be used to predict future values for the time series, this is important in for instance sales forecasting.
Within Assembléon time series analysis would be interesting to apply on the user profile. Before this is possible first the user profile has to be transformed so time series analysis could be applied. The data has to be registered over time as described in section 8.2.5. Time series analysis could be used to identify the relation that possibly exists between variables like change of hardware or change in placement program and a reliability measure like failures that occur. Identifying this relation could help with the prediction of failures.
Another possibility could be to use time series analysis to identify relationships that possibly exist between several variables. Identifying these relationships could be useful when constructing a neural network. The amount of factors included in a neural network has influence on the sample size that is needed to train and validate the neural network. As described in section 5.3 factors can be related to each other. Being able to identify this relationship makes it possible to select one variable to represent the other, thereby reducing the amount of factors that have to be included in the neural network.

9. Recommendations

As mentioned throughout the report and in the conclusion the small data set influences strongly the number of factors that can be included in the neural networks and the success of using a neural network to predict the reliability performance. The collection of more data is therefore the most important step in a follow up research that wants to use neural networks as prediction method, but probably also when other methods are used.

Increasing the sample size requires the development of a system that automatically can gather necessary information from machines all over the world. After the collection of information the analysis of data should also be performed automatically. The automatic data collection and analysis does not only make it possible to gather larger amounts of data and save time during analysis, it also increases the accuracy and the reliability of the data collection and analysis. By increasing the accuracy less data points have to be deleted because not all data is available.

The use of MTBF as reliability measure for an Assembléon machine posses some problems as described in section 8.2.4. For future research it would be interesting to investigate whether the use of other reliability measures would solve some of the issues, like the large time period required, that the use of MTBF introduces. One of the possibilities could be to just use the number of failures in stead of the MTBF. This would make it possible to collect information about variables in shorter time periods, as suggested in 8.2.4. When using MTBF as reliability measure a failure has to occur in order to calculate a MTBF. When using the number of failures as reliability measure it is possible to register zero failures in a time period. Therefore the duration of the time period in which the number of failures is counted can be adjusted so it is suitable for other variables, as described in section 4.2.1.

Selecting which factors to include in a neural network is a difficult process because the researcher has to exclude factors due to practical issues while these factors could be important to include in the model. A mayor practical problem is that some factors like user skill and environment are hard to quantify. A possibility could be to gather over a large period of time information of one specific machine. When assuming that this machine is operated by the same group of operators and is stationed at the same location over time, the influence of for instance, the environment and operators on the reliability of the machine could be kept constant. When this machine shows the same fluctuations in reliability over time as several other machines do, it could be that the environment or operator has no large impact on the reliability of the machine. Therefore it is not a problem that these factors cannot be quantified.

This research used neural networks to predict the reliability of an Assembléon machine. Neural networks have as advantage that the mathematical relation between variables doesn't have to be known because the model can "learn" the relationship between variables. Besides the disadvantage of a large sample size that is needed, a neural network acts as a black box. When the model is capable of predicting the reliability it is not possible to determine exactly how strong certain factors influence the reliability. If this influence could be determined, it would be possible to determine which factors have the strongest influence on the reliability. This information could be useful during development and testing because extra attention could be paid to these factors. This extra attention could help developing a more reliable machine.

A method that can analyze the relationship between different variables is time series analysis. For future research it would be interesting to investigate if time series analysis could be applied to identify relationships that exist between variables like change in placement programs and a reliability measure like the MTBF. Identifying could be useful during development and testing as just described.

In this research the user profile is used for the construction of a data set that can be used for the training and validation of a neural network. The user profile can also be used for other purposes. Analyzing the user profile can identify functionalities that are often used. When these functionalities are identified extra attention can be paid to these functionalities during the development and testing. The extra emphasis on these functionalities could help to extend the possibilities of these functionalities or simplify the functionalities, thereby increasing the usability of the machine.
User profiles could also be used for marketing purposes. Identifying configurations of a machine that is often used to produce certain products can help advising customers, when they are planning to buy an Assembléon machine, to select an optimal configuration of the machine for the products they are planning to produce.

References

[Abd94] Abdi, H., (1994) A neural network primer, *Journal of Biological systems*, Vol 2, pp 247-283

[Abd03] Abdi, H., (2003) Neural Networks, *Encyclopedia for research methods for the social sciences Journal of Biological systems*, pp. 792-795

[And95] Anderson, J.A., (1995) An introduction to Neural Networks, *the MIT Press*

[Ans99] Ansell, J., Walls, L., Quigley, J., (1999) Achieving growth in reliability, *Annals of Operations Research*, Vol. 91, pp 11-24

[Bal08] Balasubramanian, A., Kevrekidis, K., Sonnemans, p., Newby, M., (2008) Identifying factors influencing reliability of professional systems

[Cha04] Chatfield, C., Tanner, M., Zedek, J., (2004) The analysis of time series an introduction, *Chapman & Hall/CRCC*, sixth edition

[Dem07] Demuth, H., Beale, M., Hagan, M., (2007) Neural Network Toolbox 5: User's Guide, *http://www.mathworks.com/access/helpdesk/help/pdf_doc/nnet/nnet.pdf*, Version 5.1, The MathWorks Inc., USA.

[DiR02] DiRusso, S., Chahine, A., Sullivan, T., Risucci, D., Nealon, P., Cuff, S., Savino, J., Slim, M., (2002). Development of a model for prediction of survival in pediatric trauma patients: Comparison of artificial neural Networks and Logistic Regression, *Journal of Pediatric Surgery*, Vol. 37, No 7, pp 1098-1104

[Eco04] Economou, M. (2004) The Merits and Limitations of Reliability Prediction, *Proceedings of Annual Reliability and Maintainability Symposium,* pp. 352-357.

[Fra08] Franck, P. (2008) Preparation Master thesis, Early reliability prediction.

[Goe85] Goel, A., (1985) Software Reliability Models: Assumptions, Limitations, and Applicability, *IEEE Transactions on software engineering*, Vol SE-11, No 12, pp 1410-1423

[Hai05] Hair, J., Black, C., Babin, B., Anderson, R., Tathan, R., (2005) Multivariate Data Analysis, *Pearson Prentice Hall*, Sixth edition, ISBN 0-13-032929-0

[Jon02] Jones, J., Chong, L., Huang, J., Marshall, J., (2002) Reliability modeling and prediction using neural networks, *The 8th ISSAT International Conference on Quality and Reliability in Design*, ISBN 0963999877 pp.40-44

[Kap90] Kaplan, S., Cunha, G., Dykes, A., Shaver, D., (1990) A Bayesian Methodology for Assessing Reliability During Product Development, *Proceedings Annual reliability and maintainability Symposium*, pp. 205-209.

[Ker98] Kerscher, W., Booker, J., Bement, T., Meyer, M., (1998) Characterizing Reliability in a Product / Process Design-Assurance Program, *IEEE 1998 Annual Reliability and Maintainability Symposium*, pp 105 -112

[Ker07] Kerstens, J., (2007) Graduation project Early Reliability Comparison @ Assembléon

[Lee89] Lee, D., (1989) Improving Design Assessment Thru Improving Reliability Prediction Tools, *Delicate Balance: Technics, Culture and Consequences,* pp 192-197

[Mic01] Michael C., Schatz, M., (2001) Generating Software Test Data by Evolution *IEEE Transactions on software engineering*, Vol. 27, No. 12, pp 1085-1110

[Mor92] Moregan, M., Henrion, M., (1992) Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis, *Cambridge University Press*, ISBN 0-521-42744-4.

[Pap96] Papadrakakis, M., Papadopoulos, V., Lagaros, D. (1996) Structural reliability of elastic-plastic structures using neural networks and Monte Carlo simulation, *Computer Methods in Applied Mechanics and Engineering*, Vol. 136, issue 1-2, pp 145-163.

[Raj05] Rajpal, P., Shishodia, K., Sekhon G., (2005) An artificial neural network for modeling reliability, availability and maintainability of a repairable system, *Reliability Engineering and System Safety*, Vol. 91, pp 809–819

[Ros98] Rosenberg, L., Hammer, T., Shaw, J., (1998) Software Metrics and Reliability, NASA Satc, *9th International Symposium*

[Ros58] Rosenblatt, F. (1958) The perceptron: A probabilistic model for information storage and organization in the brain, *Psychological Review*, Vol. 65 Nr.6, pp 386-408

[Sar01] Sargent, D.J., (2001) Comparison of Artificial Neural Networks with Other Statistical Approaches Results from Medical Data Sets, *CANCER Supplement,* Vol. 91, Nr 8 pp. 1636-1642

[Siu98] Siu, N., Kelly, D., (1998) Bayesian parameter estimation in probabilistic risk assessment, *Reliability Engineering and System Safety*, pp. 89-116.

[Tu96] Tu, J., (1996) Advantages and disadvantages of using artificial neural network versus logistic regression for predicting medical outcome, *Journal of clinical epidemiology*, Vol. 49, Nr. 11, pp 1225-1231.

[Wal06] Walls, L., Quigley, J., Marshall, J. (2006) Modeling to Support Reliability Enhancement During Product Development With Applications in the U.K. Aerospace Industry, *IEEE Transactions on Engineering Management*, Vol. 53, Nr. 2, pp. 263-274

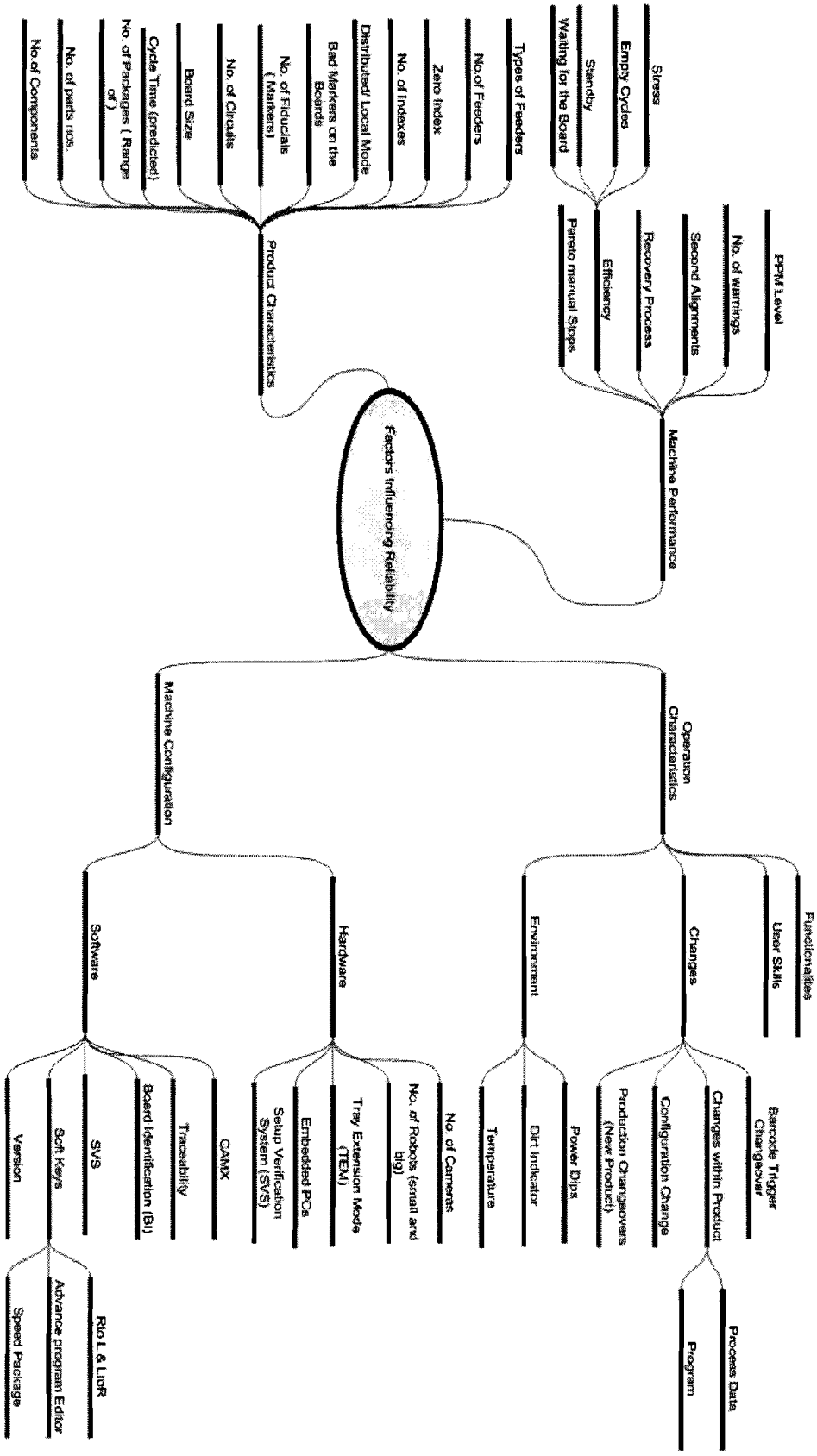
[Wid60] Widrow, B., Hoff, M. (1960) Adaptive switching circuits, *1960 IRE WESCON convention record*, vol. 4, pp 96-104.

[Won90] Wong, K.L. (1990) What is wrong with the existing reliability prediction methods?, *Quality and reliability engineering international*, Vol. 6, pp.251-257.

[Zha98] Zhang, G., Patuwo, B., Hu, M. (1998) Forecasting with artificial neural networks: The state of the art, *International Journal of Forecasting* Vol. 14, Issue 1, Pp 35-62

Appendices

Appendix A: Results of brainstorm session



Appendix B: Factors arranged by relative weight

Ranking	Factor	Relative Weight
1	Operational Skills	0,03064349
2	User Skills	0,02956099
3	Changes	0,02942281
4	No.of Placement Head Robots	0,02778011
5	Presence of Tray Extension Mode	0,02633002
6	Types of Feeders	0,02588495
7	Learning Skills	0,02421328
8	Error Level	0,02418552
9	Feeders	0,02400552
10	Maintenance	0,02394808
11	No.of Embedded PC's	0,02380139
12	No.of Manual Retry	0,02365439
13	Software Version	0,02219341
14	Power dips	0,0221326
15	Production Changeover	0,02129823
16	Cleanliness	0,02042668
17	No.of Feeders	0,02021358
18	Changes within Product	0,01991604
19	Program Change	0,01991547
20	Humidity	0,0199066
21	Configuration Change	0,01964262
22	No.of Auto Retry	0,01944072
23	Use of Advance Program Editor	0,0191773
24	Product Complexity	0,01896488
25	Environment	0,01894884
26	Functions	0,01887861
27	Process Data Change	0,01886595
28	Temperature	0,01847247
29	SVS hardware	0,0180076
30	SVS hardware	0,0177238
31	Operator Trigger Changeover	0,01702753
32	Efficiency	0,01675303
33	No. of Component Alignment Cameras	0,01627586
34	No.of Tool Bits	0,01625315
35	No.of Components	0,01581728
36	Fiducials	0,0143838
37	Bad Markers on the Board	0,01419592
38	No.of Recovery Manager Retry	0,01383631
39	No.of Fiducials	0,01372119
40	No.of Packages	0,01359269
41	No.of Second Alignment Retry	0,01351043
42	Board Size	0,0125034
43	Distributed/ Local Mode	0,01248985

44	No.of Part nos	0,01234662
45	Pareto Manual Stop	0,01220222
46	Barcode Trigger Changeover	0,0115728
47	Multi-Language	0,01086647
48	Cycle Time	0,0108222
49	Speed Package 1 & 2	0,01048565
50	Board Indentification	0,01045272
51	Traceability	0,00996284
52	No.of Circuits	0,00922776
53	No.of Indexes	0,00807894
54	Transport Indexing	0,00792036
55	Data Interface	0,00779123
56	LtoR RtoL Transport movement	0,00748906
57	SoftKeys	0,0068151
58	Use of Zero-Indexing	0,00604962

Appendix C: Factors included in the user profile

Main category	Sub-category	factor
Machine configuration	Installation	Region / country
	Standard	Base type (AX3/AX5)
		# of placement modules (PC+PR+PH)
	Hardware configuration detailed	# of CPR's / SPR's
		# of Prodrive / Advantech PC's
		# of PH-LV / PH-SV
		# of empty robot places
		# of component alignment cameras
	Hardware / software configuration, options	Tray present (y/n)
		SVS present (y/n)
		BI enabled (y/n)
		Transport direction (L2R or R2L)
		Speedpack 1/2
	Software status	Application SW version
Operation Characteristics	Change-overs	# of CO's (per week/month)
		# programs run / week
		# of different programs run / week
	Environment	(Environmental) temperature
		Average XY Temperature
		Average PH Temperature
Product Characteristics	Board	Dimensions
		# of circuits
		# of PA's
		# of packages
		# of components / board
		# of fiducials / board (measurements!)
		% CA align placements (RAM data!)
	Transport	# of indices
		Local / DSF mode
		Zero index (y/n)
		Calculated cycle time
	Feeders	# of feeder lanes used
		Type of feeders used (range)
		# of TTF
		# of ITF <= 16mm
		# of ITF > 16mm
		Tray feeder
	Toolbits	# of toolbit actions

		% CPL placements
Machine Performance	Pick&Align performance	PPM_AutoRepickAfterPickError
		PPM_AutoRepickAfterAlignError
		PPM_ManualRecoverAfterPickError
		PPM_ManualRecoverAfterAlignError
		2nd alignment attempts (ppm)
	Manual stops	# of assists (ppm)
	Efficiency	% productive time (wrt total time)
		% down time (wrt total time)
		% standby time (wrt total time)
		average recovery time assists [s]
		# pick attempts
		total time [s]

Appendix D: Coding used for nominal variables in the regression analysis

Factor	value	code
Region	Europe	1
	Asia	2
	South America	3
Base type (AX3/AX5)		0 = AX3 1 = AX 5
Prodrive / Advantech PC's		0 = Prodrive PC 1 = Advantech PC
SVS present (y/n)		0 = no 1 = yes
Speedpack 1/2	none	0
	speedpack 1	1
	speedpack 2	2
Application SW version	milestone version 3.10	3.1
	milestone version 3.20	3.2
	milestone version 3.30	3.3

Appendix E: Results of the regression analysis

Machine configuration

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	,609[a]	,371	,209	160,155

a. Predictors: (Constant), software, SVS, Region, Speedpack, Base, SPR, Controler, CPR

ANOVA[b]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	469957,001	8	58744,625	2,290	,047[a]
	Residual	795137,398	31	25649,593		
	Total	1265094,400	39			

a. Predictors: (Constant), software, SVS, Region, Speedpack, Base, SPR, Controler, CPR

b. Dependent Variable: MTBF

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.	Correlations			Collinearity Statistics	
		B	Std. Error	Beta			Zero-order	Partial	Part	Tolerance	VIF
1	(Constant)	393,796	293,647		1,341	,190					
	Region	-8,022	57,859	-,023	-,139	,891	,050	-,025	-,020	,711	1,406
	Base	-175,942	192,381	-,376	-,915	,367	-,403	-,162	-,130	,120	8,333
	CPR	8,252	21,974	,335	,376	,710	,154	,067	,053	,025	39,285
	SPR	6,461	45,385	,116	,142	,888	-,333	,026	,020	,031	32,532
	Controler	-89,798	75,733	-,250	-1,186	,245	-,377	-,208	-,169	,457	2,186
	SVS	-20,464	75,158	-,050	-,272	,787	-,302	-,049	-,039	,605	1,652
	Speedpack	-120,960	81,996	-,254	-1,475	,150	-,278	-,256	-,210	,681	1,468
	software	33,988	40,227	,131	,845	,405	,216	,150	,120	,844	1,184

a. Dependent Variable: MTBF

Machine performance

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	,480[a]	,230	,044	171,4551

a. Predictors: (Constant), Pickattempts, Downtime, Autorepickafteralignerror, Manualrecoveryafterpickerror, Manualrecoveryafteralignerror, Autorepickafterpickerror, Standbytime, productivetime

ANOVA[b]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	290070,056	8	36258,757	1,233	,311[a]
	Residual	970095,582	33	29396,836		
	Total	1260165,638	41			

a. Predictors: (Constant), Pickattempts, Downtime, Autorepickafteralignerror, Manualrecoveryafterpickerror, Manualrecoveryafteralignerror, Autorepickafterpickerror, Standbytime, productivetime

b. Dependent Variable: MTBF

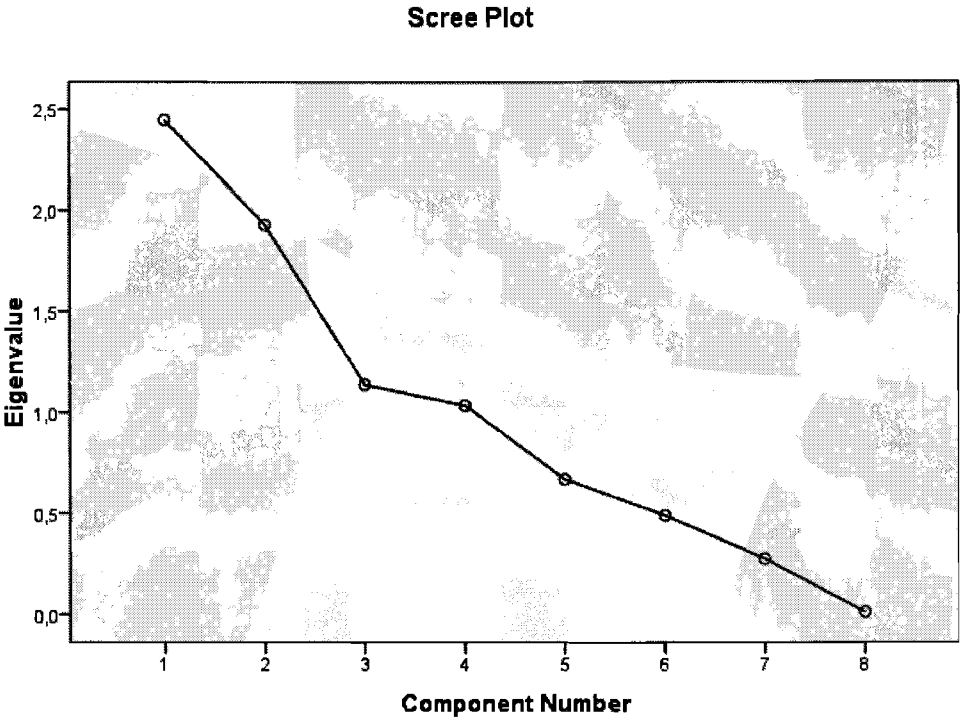
Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.	Correlations			Collinearity Statistics	
		B	Std. Error	Beta			Zero-order	Partial	Part	Tolerance	VIF
1	(Constant)	294,279	58,601		5,022	,000					
	Autorepickafterpickerror	,005	,020	,207	,259	,797	-,091	,045	,040	,037	27,181
	Autorepickafteralignerror	-,002	,009	-,172	-,180	,858	-,074	-,031	-,028	,026	39,203
	Manualrecoveryafterpickerror	-,183	,208	-,657	-,881	,385	-,183	-,152	-,135	,042	23,855
	Manualrecoveryafteralignerror	,420	,531	,565	,792	,434	-,122	,137	,121	,046	21,820
	productivetime	154,290	100,304	8,195	1,538	,134	,000	,259	,235	,001	1216,638
	Downtime	-172,167	324,631	-2,828	-,530	,599	-,013	-,092	-,081	,001	1218,853
	Standbytime	-310,929	141,504	-5,377	-2,197	,035	-,029	-,357	-,336	,004	256,720
	Pickattempts	-1,304E-6	,000	-,213	-1,220	,231	,038	-,208	-,186	,768	1,302

a. Dependent Variable: MTBF

Appendix F: Results of the factor analysis

Machine configuration



Component Matrix[a]

	Component			
	1	2	3	4
Region	,643	,154	-,299	,148
Base	,262	,608	,663	-,033
CPR	,915	-,068	,361	,022
SPR	-,892	,314	-,074	-,003
Controler	,149	,875	-,226	,001
SVS	-,085	,774	-,116	,322
Speedpack	-,551	-,051	,638	,209
software	,035	-,257	-,041	,931

Extraction Method: Principal Component Analysis.

a. 4 components extracted.

Rotated Component Matrix[a]

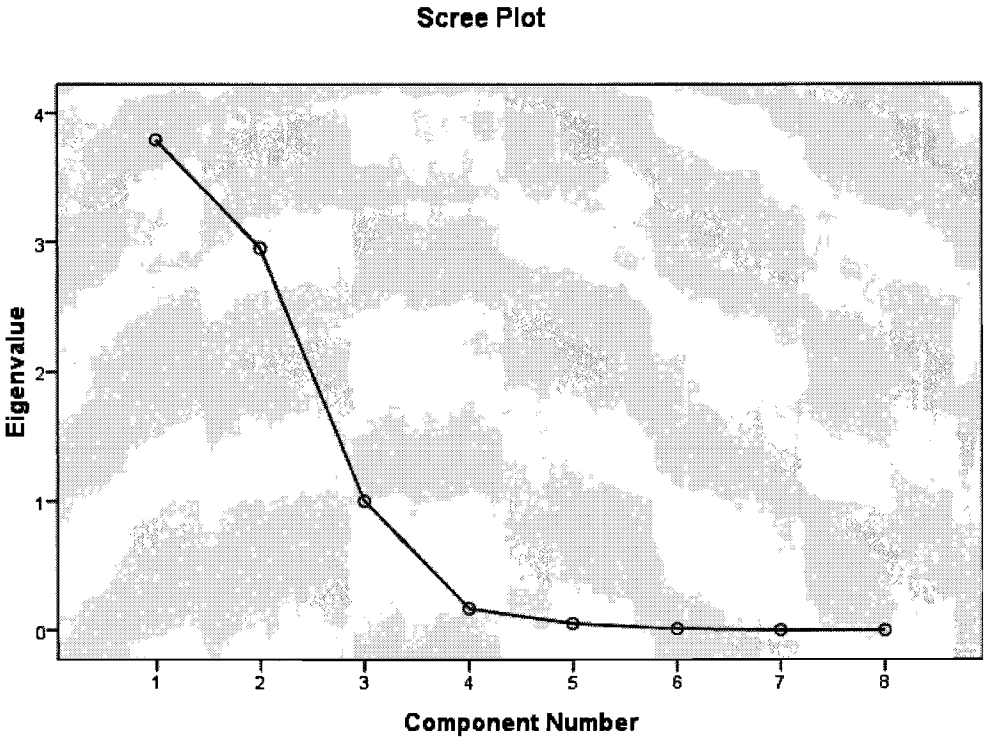
	Component			
	1	2	3	4
Region	,426	,235	-,539	,146
Base	,515	,569	,478	-,245
CPR	,979	-,021	-,111	,007
SPR	-,835	,254	,364	-,079
Controler	-,011	,867	-,232	-,179
SVS	-,151	,825	,006	,140
Speedpack	-,178	-,061	,838	,139
software	,054	-,030	,044	,964

Extraction Method: Principal Component Analysis.

Rotation Method: Varimax with Kaiser Normalization.

a. Rotation converged in 6 iterations.

Machine performance



Component Matrix[a]

	Component		
	1	2	3
Autorepickafterpickerror	,943	,184	,074
Autorepickafteralignerror	,959	,189	,075
Manualrecoveryafterpickerror	,951	,189	-,053
Manualrecoveryafteralignerror	,949	,173	-,012
productivetime	-,247	,969	,018
Downtime	-,240	,970	,007
Standbytime	-,236	,971	-,016
Pickattempts	-,078	-,024	,996

Extraction Method: Principal Component Analysis.

a. 3 components extracted.

Rotated Component Matrix[a]

	Component		
	1	2	3
Autorepickafterpickerror	,963	-,023	,033
Autorepickafteralignerror	,979	-,021	,033
Manualrecoveryafterpickerror	,967	-,020	-,095
Manualrecoveryafteralignerror	,962	-,036	-,054
productivetime	-,032	,999	,015
Downtime	-,025	,999	,003
Standbytime	-,023	,999	-,020
Pickattempts	-,038	-,002	,998

Extraction Method: Principal Component Analysis.

Rotation Method: Varimax with Kaiser Normalization.

a. Rotation converged in 3 iterations.

Appendix G: Results of the regression analyses reduced amount of variables

Machine configuration

Variables Entered/Removed[b]

Model	Variables Entered	Variables Removed	Method
1	Speedpack, software, Controler, CPR[a]	.	Enter

a. All requested variables entered.

b. Dependent Variable: MTBF

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	,558[a]	,311	,233	157,764

a. Predictors: (Constant), Speedpack, software, Controler, CPR

ANOVA[b]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	393963,606	4	98490,902	3,957	,009[a]
	Residual	871130,793	35	24889,451		
	Total	1265094,400	39			

a. Predictors: (Constant), Speedpack, software, Controler, CPR

b. Dependent Variable: MTBF

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	403,572	130,133		3,101	,004
	CPR	,952	3,618	,039	,263	,794
	Controler	-156,883	52,575	-,436	-2,984	,005
	software	43,283	37,040	,167	1,169	,250
	Speedpack	-179,746	71,686	-,378	-2,507	,017

a. Dependent Variable: MTBF

Machine performance

Variables Entered/Removed[b]

Model	Variables Entered	Variables Removed	Method
1	Pickattempts, Downtime, Autorepickafteralignerror[a]	.	Enter

a. All requested variables entered.

b. Dependent Variable: MTBF

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	,084[a]	,007	-,071	181,4638

a. Predictors: (Constant), Pickattempts, Downtime, Autorepickafteralignerror

ANOVA[b]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	8859,202	3	2953,067	,090	,965[a]
	Residual	1251306,436	38	32929,117		
	Total	1260165,638	41			

a. Predictors: (Constant), Pickattempts, Downtime, Autorepickafteralignerror

b. Dependent Variable: MTBF

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	235,884	41,406		5,697	,000
	Autorepickafteralignerror	,000	,001	-,074	-,457	,650
	Downtime	-,993	9,853	-,016	-,101	,920
	Pickattempts	2,252E-7	,000	,037	,227	,822

a. Dependent Variable: MTBF