December 15, 2016

MASTER ASSIGNMENT

TOWARDS A STANDARDIZED AVAILABILITY MODEL

N. Gietema BSc

Faculty of Behavioural, Management and Social sciences (BMS)
Chair, Industrial Engineering and Business Information Systems (IEBIS)

Supervisors:
University of Twente
Dr. Ir. A. Al Hanbali
Dr. Ir. L.L.M. van der Wegen

Vanderlande Industries
Ir. M. Verhoeven

Documentnumber
IEBIS — 1

VANDERLANDE® INDUSTRIES

UNIVERSITY OF TWENTE.

Management summary

Downtime of a system is expensive for the customers of Vanderlande. Therefore, availability of a system is an important key performance indicator during the design phase of a material handling system. Currently, each market segment within Vanderlande uses its own methods to calculate the availability of a system in the design phase. Moreover, the major part of the projects is done without a detailed availability study. In addition, even if a calculation is done, the calculated availability is often inaccurate, mostly resulting in a higher availability than the measured availability when the system is installed and operational at the customer. In addition, service contracts are requested more and more by the customers of Vanderlande. Therefore, knowledge of expected failure behaviour and availability of the systems are crucial in order to calculate the expected service costs in an accurate way. All this gives rise to develop a standardized model that calculates the availability in a fast and accurate way.

We design a simple time simulation model to estimate the system availability. We use simulation because it allows for a flexible model which is standardized for each business unit and for each user. Our model loops over the user defined time horizon and simulates in each time period the state of each equipment. By knowing the state of each equipment, the system state is obtained. We count the number of times the system is working and calculate the availability by dividing this number by the number of time periods. We repeat this simulation process by adding more replications of the simulation until we obtain the desired level of accuracy.

We found that our model is accurate for parallel and k-out-of-n structures. For serial structures our model underestimates system availability compared to the exact serial system availability. A combination of different structures gives accurate results. The running time exponentially increases with the number of components. This increase is even larger for simulation of parallel and k-out-of-n structures.

Our study is a first research towards standardized, fast and accurate availability calculation. This results in a prototype tool that can handle small material handling systems and estimate its system availability. We recommend to further develop our tool to make it suitable for the large and complex systems Vanderlande designs by making the software more efficient, designing a user friendly layout and by adding relevant features such as subsystem availability calculation and a list of critical components.

Table of contents

Preface

It took some time, but I am glad to present to you this research thesis which I wrote in the context of the completion of the master study Industrial Engineering & Management at the University of Twente. As expected, the research study went not without struggle. I spent more time than planned on literature research, reading irrelevant articles. Furthermore, my tendency to perfectionism might improve the final results, but makes the process of achieving it more difficult. I would like to thank Ahmad al Hanbali and Maarten Verhoeven for guiding me through the master assignment and providing advices to improve my knowledge and skills. Furthermore, I would like to thank Joep Geurts who answered a lot of questions I had concerning the way of working of Vanderlande. This thesis is about research towards a standardized availability model for material handling systems designed by Vanderlande Industries. This will be further introduced in the next chapter.

Chapter 1

Introduction

In this research we develop a model to calculate the availability of automated material handling systems. The project is done for the master thesis of the study Industrial Engineering and Management at the University of Twente. It is executed in six months at the systems department of Vanderlande Industries. This introductory chapter first describes the context of the research in Section 1.1, then formulates the problem identification in Section 1.2, followed by the problem definition and relevant research questions in Section 1.3. We describe the scope of the research in Section 1.4. The chapter ends with a description of the methodology used and the structure of the remaining chapters in Section 1.5.

1.1 Vanderlande Industries

Vanderlande Industries, hereafter referred to as Vanderlande, is a worldwide operating company founded in 1949. Their headquarters are based in Veghel, the Netherlands. They are world's fifth supplier of material handling systems (Modern Material Handling, 2015) with net sales around 790 million euros in 2014 (Vanderlande Industries, 2014). The company has circa 2800 employees and is in the top three of Dutch employers (Vanderlande Industries, 2014). Vanderlande designs, realizes and optimizes automated material handling systems and services. Figure 1.1 shows the markets which Vanderlande is active in together with their share in total sales: baggage handling, warehouse automation and parcel & postal. In addition they provide life-cycle services for their customers such as maintenance and logistic management. In the next sections we briefly describe these four business units.

1.1.1 Baggage handling

Baggage handling systems are systems for transportation, storage and sorting of baggage. Figure 1.2 shows some examples of baggage handling systems. This includes the check-in process and the transfer and arrival of baggage. Vanderlande is market leader in this segment and has installed more than 1000 baggage systems at 600 airports worldwide including Schiphol Airport, London Heathrow

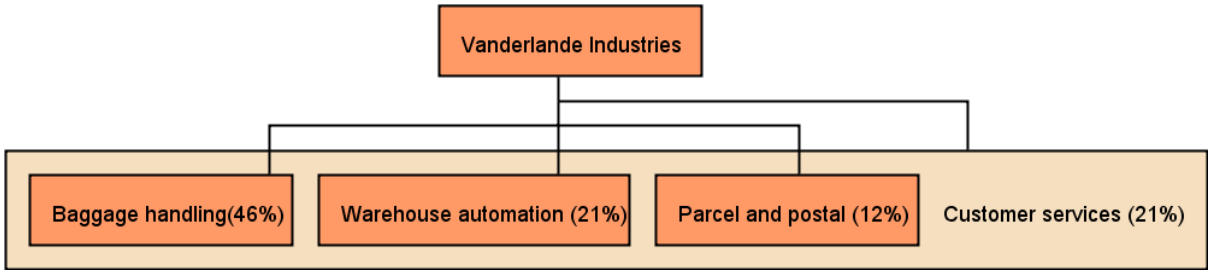


Figure 1.1: Business units (percentage of total sales) (Vanderlande Industries, 2014)

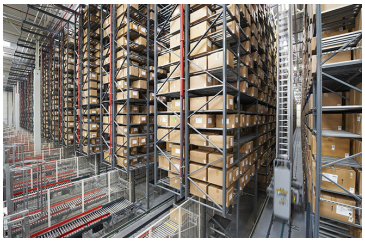
(a) Check-in system

(b) Baggage sorting system

(c) Baggage reclaim carousel

Figure 1.2: Example of baggage handling systems

(a) Goods-to-man orderpick system



(b) Automatic storage & retrieval

(c) Automatic case picking system

Figure 1.3: Example of warehouse automation systems

Airport and Vancouver Airport (Vanderlande Industries, 2015a). With increasing passenger volumes, the baggage handling market is a competitive growing market with the highest growth in the Middle East, India and China (Vanderlande Industries, 2014). Market experts expect that in the coming years the number of flight connections will increase together with shorter connection times (Vanderlande Industries, 2014). Therefore, airports will demand fast and reliable baggage handling systems more than ever before.

1.1.2 Warehouse automation

Vanderlande is one of the three largest players in the warehouse automation and distribution market. The company provides solution for automated warehouses and distribution centres using automated storage and retrieval systems and systems for order picking, cross docking and sorting (Vanderlande Industries, 2015d). Figure 1.3 shows three examples of warehouse automation systems. Vanderlande is active in four areas. First, Vanderlande has customers in the fashion industry such as H&M and Nike. Food retail is the second area with customers such as Tesco and Carrefour. Thirdly, they provide solutions for automated transport and storage for parts & components. For example, they installed systems for Bosch and Arrow Electronics. Finally, they design and deliver systems for the e-commerce business with large customers such as Amazon and Zalando (Vanderlande Industries, 2015d). In these growing market areas there is a need to reduce logistic costs and for faster order delivery (Vanderlande Industries, 2014). It is a challenging task of Vanderlande to come up with new innovative systems that meet the customer's needs.

1.1.3 Parcel & postal

With more than 500 customers, Vanderlande is market leader in the parcel & postal industry. The company provides fully automated parcel sorting centres including systems for receiving, screening, sorting and shipping of mail and parcels (Vanderlande Industries, 2015c). Figure 1.4 shows some examples of parcel systems. Vanderlande has installed systems for FedEx, DHL, UPS and Austria Post (Vanderlande Industries, 2014). As for the warehouse automation industry, the parcel & postal industry demands fast and reliable delivery together with cost reductions due to the growing e-commerce

(a) Screening of parcels

(b) Parcel sorting

(c) Shipping of parcels

Figure 1.4: Example of parcel and postal systems

business (Vanderlande Industries, 2014). Therefore, for this market segment, as well as for the others, Vanderlande's systems are becoming more complex due to more advanced technologies, higher throughput and larger utilization.

1.1.4 Customer services

Vanderlande offers a wide range of services to their customers. They can take care of preventive and corrective maintenance in different degrees. Site-based maintenance is possible as well as assistance via hotlines and help desks. Furthermore, Vanderlande can take over spare part management. In addition, the company can provide training about maintenance, equipment usage and safety. Moreover, life-cycle plans, logistics consultancy and process monitoring are services offered by Vanderlande as well. Finally, customers have the option to actively take part in the design process of their system by becoming a business partner (Vanderlande Industries, 2015b). All these customer services can be provided via custom made contracts.

1.2 Problem motivation

Downtime of a system is expensive for the customers of Vanderlande. For example, suppose the baggage handling system at a large airport breaks down. This has tedious and costly consequences such as passengers who are getting their luggage late. Therefore, availability of a system is an important key performance indicator during the design phase of a material handling system. Currently, each market segment within Vanderlande uses its own methods to calculate the availability of a system in the design phase. Moreover, the major part of the projects is done without a detailed availability study. In addition, even if a calculation is done, the calculated availability is often inaccurate, mostly resulting in a higher availability than the measured availability when the system is installed and operational at the customer. Since there is no standardized model, calculations can differ between market segments and even among projects in the same segment. In addition, service contracts are requested more and more by the customers of Vanderlande. Therefore, knowledge of expected failure behaviour and availability of the systems are crucial in order to calculate the expected service costs in an accurate way. Performing an availability study during the design phase can provide this knowledge. Finally, the availability calculations are done manually. All this gives rise to develop a standardized model that calculates the availability in a fast and accurate way.

Before we state our research objective, we first describe the definition of availability that we use in this study.

1.2.1 Definition of availability

There are several ways to define availability, each definition with a slightly different interpretation. For example, Rausand & Hoyland (2004) define availability (according to the BS4778 quality vocabulary) as *'the ability of an item (under combined aspects of its reliability, maintainability and maintenance support) to perform its required function at a stated instant of time or over a stated period of time.'* This is a precise and complete theoretical definition. Vanderlande, however, uses a more practical definition as described by the European Federation of Materials Handling (1989):

'availability is the probability of finding a system at a given time in an operable condition.'

The complement of availability is unavailability which can be divided in two parts: operational and technical unavailability. The first consists of downtime due to improper use of the system. For example, failures due to transport of products that are outside the allowed product dimensions fall within operational downtime. Technical unavailability includes downtime due to technical failures such as worn out parts or software bugs. Vanderlande is only responsible for the technical availability of the system. It is the customer's responsibility to use the system in the right way and the customer is therefore accountable for operational availability. Vanderlande and the customer together agree on the nature of the failure. This can partially be done in advance by making a list of possible failures and their nature. If not listed failures happen, Vanderlande and its customer will agree if it is a technical or operational failure.

To come back to Vanderlande's definition of availability, it is not exactly clear what is meant by *'an operable condition'*. We define, using the definition of Vlasblom (2009), a system to be in operable condition when the system can meet the throughput of baggage, parcels or materials, where Vanderlande and its customer agreed on. To be a bit more precise, we define technical unavailability as follows:

a system is unavailable when due to technical failure the system can not meet the throughput where Vanderlande and its customer have agreed on' (Vlasblom, 2009).

With throughput we mean the required hourly peak flow of material handling units. For baggage handling, the throughput will be in terms of baggages per hour, whereas this is parcels per hour for the parcel & postal section. For warehouse automation, this can be for example pallets or cases per hour. To keep our model simple, we will focus on average availability over a long-term. However, Vanderlande will be penalized if the technical availability is below the agreed threshold. Therefore we prefer a model that can compute the availability for a specific time interval as well.

1.3 Research objective & questions

As discussed in the previous section, the aim of the study is to develop a standardized model for the technical availability calculation. Furthermore, we want to know how we can encourage people working at Vanderlande to make use of this model. We therefore formulate the following research objective:

Research towards a usable standardized model for calculating technical avaibility of automated material handling systems during the design phase.

To develop such a model we first need to know in what way Vanderlande currently does their availability calculations, what the differences are between the market segments and how the availability study is used. Next, we need to know which factors influence the technical availability of a system and what basic models for availability calculation exist. If we know the possible approaches and we know the requirements of Vanderlande regarding the to be developed model, we can choose a modelling approach. Last but not least, we can develop the model and implement it in prototype. Therefore, we formulate the following five research questions which are explained below:

1. Which models are suitable for calculating technical availability and what parameters are needed for it?

For this first question we study the literature and we interview several people within Vanderlande who are involved with the availability calculation. We describe the factors that significantly influence the system's technical availability. Furthermore, we describe several methods to calculate availability and discuss their advantages as well as their disadvantages.

2. How is the current situation regarding availability calculations?

To answer the second research question we first study how the availability calculation is used within Vanderlande. Furthermore, we describe what model Vanderlande uses for each market segment. Third, we discuss the strengths and weaknesses of the currently used methods. Finally, we explain how Vanderlande measures availability of operational systems.

3. What is the most appropriate modeling approach for availability calculation for Vanderlande?

Regarding the third question, to know what modeling approach is most suitable, we first need to know Vanderlande's requirements for availability calculation. Next, we can weigh the alternatives and select the best modeling approach.

4. How can the chosen approach be applied at Vanderlande in order to develop the model?

5. How can the model be tested and validated?

By describing our availability model we answer the fourth and research question. We explain the input parameters we use and describe the calculation method. Finally, we discuss the testing and validation of our prototype model.

In the next section we describe the scope of our research.

1.4 Scope

In this section we discuss the scope of our research, because time for this research is limited. We describe which aspects we take into account and which are excluded from the research. Currently, Vanderlande calculates availability based on equipment. Using formulas for serial and parallel structures availability on higher levels is calculated. We base our model on the equipment level as well, because Vanderlande keeps track of the status of their equipment and thus availability data on equipment level is available within Vanderlande. Next, we do not focus on collecting input data. Collecting input data is not straightforward and will require too much time. However, we describe what input parameters are needed to take into account in the next chapter. We assume that spare parts are always available. Furthermore, because of the limited time available we do not take into account the maintenance policy.

Finally, the input data is outdated. Many of the component availability figures are not updated for years and it is not precisely known how these figures are obtained.

The remaining section of this chapter describes the methodology used to answer the stated problem definition and research questions and outlines briefly the content of the rest of the report.

1.5 Methodology and structure of the report

In order to do structured research we use the Managerial Problem Solving Method (MPSM) (Heerkens & Van Winden, 2012). The MPSM is a common sense based, generally applicable and systematic approach taking into account the context of the organization in order to generate solutions that fit the

company. The main steps are identification of problems, analysis of the core problem, designing solutions, implementation of the chosen solution and evaluation of the results.

The problem identification is already set out in this introductory chapter. We outline the relevant literature regarding system availability in Chapter 2. In Chapter 3 we describe the current method for availability calculation. Then, in Chapter 4, we describe our modelling approach. In Chapter 5, we develop a technical availability calculation model. In the subsequent chapter we apply our model to an example system for each market segment. We omit the implementation and evaluation step of the MPSM because the model is not implemented yet. The report ends with a conclusion and discussion of the study and some recommendations to further improve the accuracy of the availability calculation and the way of using it.

Chapter 2

Theoretical framework

In this chapter we answer our first research question: which models are suitable for calculating technical availability and what parameters are needed for it? We outline the relevant literature regarding availability of a system. We start with discussing factors that influence the system availability in Section 2.1. Next, we describe four main models to model and calculate the availability. We end this chapter with a description of the component importance in Section 2.3.

2.1 Factors influencing system availability

This section discusses two main areas that have impact on system availability. The first area is the availability of the individual components. The higher the components' availability, the higher the availability of the system (Kuo & Wan, 2007). Second, the structure of the system has large impact on availability as well. (Kuo & Wan, 2007). We discuss the influence of component availability on system availability in Section 2.1.1 and the influence of the structure of the system in Section 2.1.2.

2.1.1 Component availability

Component availability is an important determinant of system availability. In general, enhancing component availability comes with an increase in costs. Furthermore, not all components are of equal importance. For example, consider a system with three components. The first two parts are in parallel, whereas the third component is in series with the parallel structure. Figure 2.1 shows this system. Suppose all three components have equal availability. Enhancing the availability of the third component will have a larger impact than improving the availability of one of the other two components. To reach a required system availability it is important to compute the optimal availability of each component that minimizes the total cost (Mettas, 2000). There are several methods to solve this optimization problem. However, we will not discuss them here since this is out of our project scope. Nevertheless, identifying critical components can be useful during the design phase of a system. Section 2.3 describes how to calculate the importance of a component.

The availability of a component depends on the failure rate of the component and the repair rate of the component. The failure rate is the frequency with which the component fails. We express this in failures per time unit. This failure rate has a certain probability distribution such as an exponential or Weibull distribution. If we know the failure rate we can compute the mean time to failure (MTTF). The repair rate is the frequency with which the system will be repaired once a failure occurred. Like the failure rate, the repair rate has a certain distribution. Furthermore, we can compute the mean time to repair (MTTR). Equation 2.1 shows the relation between availability, MTTF and MTTR:

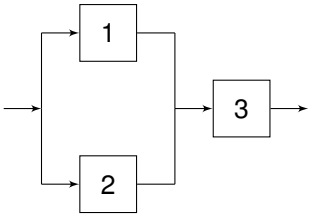


Figure 2.1: Example system

$$A_i = \frac{MTTF_i}{MTTF_i + MTTR_i}, \tag{2.1}$$

where A_i is the availability of component i and $MTTF_i$ and $MTTR_i$ are its mean time to failure and mean time to repair.

The mean time to failure is closely related to the quality and the load of the component The mean time to repair is mainly dependent on the nature of the failure, spare part logistics (Mettas, 2000), manpower for repair and its (corrective and preventive) maintenance policies (Topuz, 2009).

2.1.2 System structure

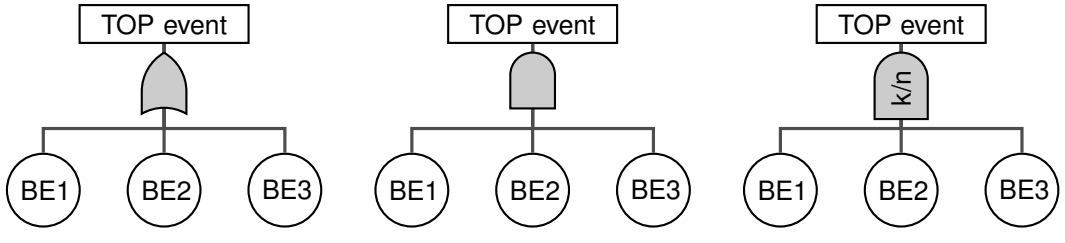
With respect to the structure of the system there are two aspects that have an impact on the system availability. The level of redundancy is important. With redundancy we mean components that are in parallel. The strongest level of redundancy is a pure parallel structure for which only one of the components needs to work. Lower levels of redundancy are structures for which only some of the components need to work. For example, if multiple components are placed in series, the system is down if one of the components fails. In contrast, for a parallel structure only one of the components needs to work to have a working system. The second aspect enhancing the system availability is reassignment of interchangeable components (Kuo & Prasad, 2000).

2.2 Availability calculation models

In this section we describe four methods to model system availability which can be found in literature. First we discuss the failure tree analysis method, followed by a description of the reliability block diagram method. Next, we give an explanation of how Markov models can be applied to compute the availability of a system. We end this section with a description of simulation techniques to calculate system availability.

2.2.1 Failure tree analysis

Failure tree analysis or fault tree analysis (FTA) is one of the most common methods for risk and reliability studies (Rausand & Hoyland, 2004). A fault tree is a logic diagram that shows the interrelationship between possible component failures (events) and its causes in a system (Rausand & Hoyland, 2004). The diagram starts with a TOP event denoting system failure. The events that possibly can cause system failure are placed below the TOP event. Events can be connected via an OR or an AND gate. The OR gate states that only one of the connected events needs to occur in order to have a system failure. The components are serial and all the components need to work in order to have a working system. In contrast, to have a system failure with an underlying AND gate, all the connected events need to occur. This represents components in parallel. Only one of them needs to work to have a working system. A k out of n system is a system for which more than one, say k with $k > 1$, working components are



(a) OR gate (b) AND gate (c) k out of n gate

Figure 2.2: Failure tree gate representations

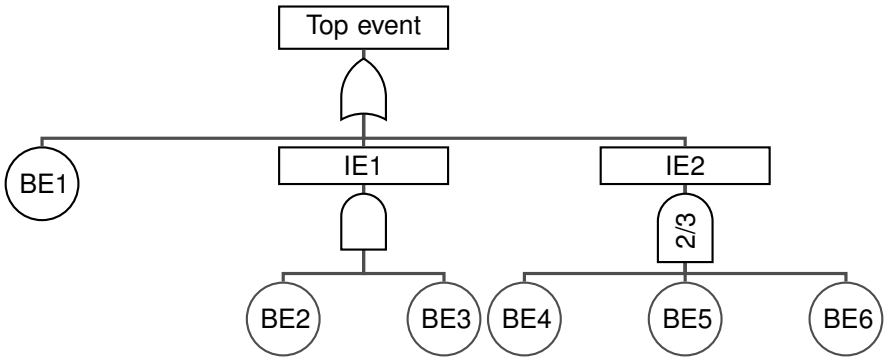


Figure 2.3: Fault tree example (Rausand & Hoyland, 2004)

required in a parallel structure. Figure 2.2 show the graphical representation of the OR gate, AND gate, and k out of n gate, respectively.

Several layers can be added to the fault tree until the desired level of detail is achieved. Figure 2.3 shows an example of a fault tree.

The fault tree can be analysed qualitatively or quantitatively. Since in this research we develop a quantitative model we only discuss the quantitative analysis. We use the approach described by Rausand & Hoyland (2004).

First we define variables describing the state of the basic events. These are the events on the lowest level of the tree:

$$Y_i(t) = \begin{cases} 1 & \text{basic event } i \text{ occurs at time } t \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

The probability that basic event i occurs at time t is denoted by $q_i(t)$, i.e. $q_i(t) = P(Y_i(t) = 1)$. This probability can be seen as the unreliability of basic event i. Let n represent the number of basic events in the fault tree. The state variables can be represented by a state vector $Y(t) = (Y_1(t), Y_2(t), \ldots Y_n(t))$. Next, the state of the TOP event is denoted by:

$$\psi(Y(t)) = \begin{cases} 1 & \text{top event occurs at time } t \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

$Q_0(t)$ is the probability that the TOP event occurs at time t, i.e. $Q_0(t) = P(\psi(Y_i(t)) = 1)$. Thus, $Q_0(t)$ represents the unreliability of the system.

As described above, components in parallel can be modelled with an AND gate. Equation 2.4 shows the formula of the state of the TOP event. The unreliability of the system if all basic events occur independently of each other is given by equation 2.5:

$$\psi(Y(t)) = \prod_{i=1}^{n} Y_i(t) \tag{2.4}$$

$$Q_0(t) = \prod_{i=1}^{n} q_i(t). \tag{2.5}$$

Serial components can be modelled with an OR gate. The state of the top event is given by equation 2.6. Assuming independent events, equation 2.7 shows formula for the unreliability of a serial structure:

$$\psi(Y(t)) = 1 - \prod_{i=1}^{n} (1 - Y_i(t)) \tag{2.6}$$

$$Q_0(t) = 1 - \prod_{i=1}^{n} (1 - q_i(t)). \tag{2.7}$$

Third, k out of n structures can be modelled with a special AND gate. If the components are identical, we can compute the unreliability using the binomial distribution (Rausand & Hoyland, 2004):

$$\psi(Y(t)) = \begin{cases} 1 & \sum_{i=1}^{n} Y_i(t) \geq k \\ 0 & \sum_{i=1}^{n} Y_i(t) < k. \end{cases} \tag{2.8}$$

$$Q_0(t) = \binom{n}{y} \sum_{y=k}^{n} q(t)^y (1 - q(t))^{n-y}. \tag{2.9}$$

Consider the example shown by Figure 2.3. Suppose the components have unreliabilities as shown by Table 2.1. Furthermore, assume the components are independent and that the components 4, 5 and 6 are identical. We start with calculating the unreliability of the first intermediate event (IE1). This event occurs if both components 2 and 3 are not working. They are in parallel and we use the formula for parallel structures. Intermediate event 2 (IE2) is a 2 out of 3 structure. Since the components are identical we can use Equation 2.8:

$$Q_{IE1}(t) = \prod_{i=2}^{3} q_i(t) = 0.2^2 = 0.04. \tag{2.10}$$

$$Q_{IE2}(t) = \binom{3}{y} \sum_{y=2}^{3} q(t)^y (1 - q(t))^{n-y} = 3 \cdot 0.3^2 \cdot 0.7 + 0.3^3 = 0.216. \tag{2.11}$$

Now we can compute the unreliability of the system. Basic event 1 and the two intermediate events are connected with OR gate, i.e. they are in series. Hence, we use the formula for a series structure:

$$Q_0(t) = 1 - \prod_{i=1}^{3} (1 - q_i(t)) = 1 - (1 - 0.1)(1 - 0.04)(1 - 0.216) = 0.3226. \tag{2.12}$$

Calculating exact system (un)reliability can be time-consuming. Therefore we describe an approximation method according to Rausand & Hoyland (2004). The approximation uses minimal cut sets. The definition of a minimal cut set is, according to Rausand & Hoyland (2004):

A cut set in a fault tree is a set of basic events whose occurrence (at the same time) ensures that the TOP event occurs. A cut set is said to be minimal if the set cannot be reduced without loosing its status as a cut set.

Component i	Unreliability
1	0.1
2, 3	0.2
4, 5, 6	0.3

Table 2.1: Component availability of example system

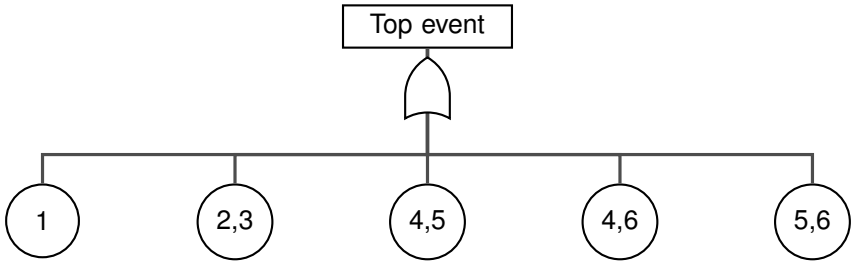


Figure 2.4: Fault tree showing minimal cuts

Generally, a minimal cut set has a parallel structure. Thus, a minimal cut set occurs if all its basic events occur. Let K denote the number of minimal cut sets of the system. Note that the same basic event can be in multiple minimal cut sets. Let $\check{Q}_j(t)$ denote the probability that minimal cut set K_j occurs. If all basic events occur independent of each other, then this probability is given by

$$\check{Q}_j(t) = \prod_{i \in K_j} q_i(t). \tag{2.13}$$

A system can be modelled with its minimal cuts sets placed in series. The TOP event, system failure, occurs if at least one of the minimal cut sets occurs. Figure 2.4 shows the fault tree with minimal cuts of the example showed by Figure 2.3. Table 2.2 shows the unreliabilities of the minimal cut sets. There are structured methods to find the minimal cuts of a system (Rausand & Hoyland, 2004).

If all minimal cuts are independent of each other, than the system failure is given by

$$Q_0(t) = 1 - \prod_{j=1}^{k} (1 - \check{Q}_j(t)). \tag{2.14}$$

However, since the same basic event can be in multiple minimal cut sets the minimal cut sets are not always independent of each other. As can be seen in Figure 2.4, component 4, 5 and 6 are in two minimal cut sets. Therefore, the minimal cut sets are not independent. If the minimal cut sets are dependent of each other, they are positive correlated. A system with serial components that are positive correlated is more reliable than a system with independent components (Rausand & Hoyland, 2004), see Equation 2.15. Moreover, if the components of a system are highly reliable we can approximate the system unreliability as if it consists of independent multiple cut sets, as shown by Equation 2.16 (Rausand & Hoyland, 2004):

Minimal cut set	Components	Unreliability $\check{Q}_j(t)$
1	1	0.1
2	2,3	0.04
3	4,5	0.09
4	4,6	0.09
5	5,6	0.09

Table 2.2: Component availability of example system

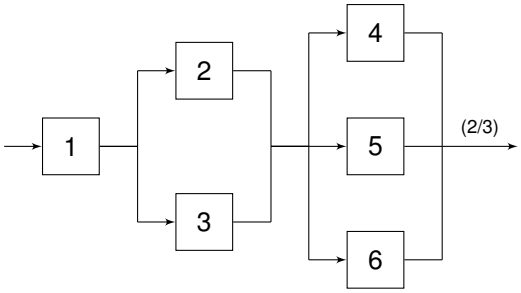


Figure 2.5: Reliability block diagram example

$$Q_0(t) \leq 1 - \prod_{j=1}^{k}(1 - \check{Q}_j(t)) \tag{2.15}$$

$$Q_0(t) \approx 1 - \prod_{j=1}^{k}(1 - \check{Q}_j(t)). \tag{2.16}$$

According to Rausand & Hoyland (2004), we need to be careful with this approximation if at least one of the unreliabilities q_i's is 0.01 or larger.

To come back to the example, since the minimal cut sets are dependent and not highly reliable, we can only compute an upper bound on the unreliability of the system:

$$Q_0(t) \leq 1 - \prod_{j=1}^{5}(1 - \check{Q}_j(t)) = 1 - (1 - 0.1)(1 - 0.04)(1 - 0.09)^3 = 0.3489. \tag{2.17}$$

Since we already know that the exact unreliability is 0.3226, this is indeed a valid upper bound.

2.2.2 Reliability block diagram

The second model we discuss is modelling the system using a reliability block diagram (RBD). Figure 2.5 shows an example of a RBD. A reliability block diagram describes the function of the system. If a system consists of more than one function, a RBD is needed for each function. A system is represented by a network of basic elements using serial and parallel structures. In contrast to the fault tree which calculates the probability of system failure, the RBD method computes the probability that the system is functioning. This method is suitable for systems of non-repairable components and for systems in which the order of failures does not matter. However, there are extensions that deal with repairable and order-sensitive failures (Rausand & Hoyland, 2004). It is possible to convert a fault tree to a RBD and vice versa. However, sometimes it is difficult to do so. If there are shared components or if the system is large or complex, then it is generally hard to compute or convert a fault tree to a RBD.

2.2.3 Markov models

In this section we describe how Markov models can be used to model availability. But first, let us briefly explain what Markov models are. Suppose for each index t (e.g. time t) there is a random variable $X(t)$ (Ross, 2007). The state of the process at t is $X(t)$. The index t often represent time. For example, $X(t)$ might be the number of working elements of a machine at time t. This can be seen as a collection of random variables and is called a stochastic process $\{X(t), t \in T\}$. Let us now consider a stochastic process that takes on a finite number of possible values, i.e. $\{X_n, n = 0, 1, 2, \ldots\}$. Suppose that if the process is in state i, there is a fixed probability P_{ij} that the next state will be j. Such a process

is called a Markov chain (Ross, 2007). Equation 2.18 shows the mathematical definition of a Markov chain. The main characteristic of a Markov chain is the memoryless property: a transition to a future state is dependent only on the current state and is independent on any of the past states.

$$P\{X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \ldots, X_1 = i_1, X_0 = i_0\} = P\{X_{n+1} = j | X_n = i\} = P_{ij} \ \forall i_0, i_1, \ldots, i_{n-1}, i, j, \text{and} n \geq 0 \tag{2.18}$$

If failure and repair times are exponentially distributed, automated material handling systems can be modelled as a Markov chain (Iyer et al., 2009). Each state can be seen as a combination of functioning and failed components. The transition to a future state represents the repair or failure of a single component. Since it does not matter how the system got into the current state, the probability of each transition is independent of the past states, but dependent on the current state. Therefore, the system can be modelled as a Markov chain.

Consider the example shown by Figure 2.5. One of the states is the situation in which all components are functioning. As a result, the system will be up. On the other hand, the state representing all components are functioning except component 1 indicates that the system will be down. For component 1 there are two states possible. For the parallel structure consisting of component 2 and 3, there are three possible states. Both components can work or can be in failure, or one of them can work. Since the two components are identical it does not matter which of them is in failure for this state. Likewise, the parallel structure consisting of components 4, 5 and 6 have four possible states. In total there are $2 \cdot 3 \cdot 4 = 24$ possible states. Only four states result in a working system. The large number of possible states shows a major disadvantage of this modelling technique. If the number of components increases, the number of states will increase exponentially. This is called the state space explosion problem (Ross, 2007). Although we can split up the system into multiple subsystems, computing the system availability will be time consuming.

2.2.4 Simulation

Simulation techniques can be used when it is difficult to use one of the previous discussed methods. Below we discuss a simple way to simulate the system and compute the availability.

Simple continuous time simulation

The simplest time simulation used for availability calculation are based on Monte Carlo simulation. Monte Carlo simulation can be defined as a scheme which makes use of random numbers to solve stochastic or deterministic problems (Law, 2007). This scheme is repeated many times and the occurrences number of specific events is counted (Takeshi, 2013). Monte Carlo simulation is a technique without time dimension. However, failure and repair rates are time-based. Therefore, a time dimension is often added to the simulation. It is broadly used for complex systems (Lu et al., 2012) that are not analytically manageable. For example, simple time simulations can easily deal with k out of n structures, redundancies, repair and maintenance (Takeshi, 2013). The simulation can be seen as a series of real experiments (Takeshi, 2013). Statistical techniques are used to estimate the mean and confidence intervals. A weak aspect of this method is that its computation time can be large. For systems with highly reliable components the sample size needs to be large to provide reliable estimates (Korver, 1994).

Consider again the example system depicted in Figure 2.5. We assume that each component has constant failure and repair rate. Since we are interested in average availability, we take a time horizon of one year. Suppose we want to obtain a 95%-confidence interval with a maximum relative error of 0.01. This means that with a probability of 0.95 the availability falls within our interval. Furthermore, we

State	Availability	Lowerbound	Upperbound
IE1	0.9590	0.9539	0.9641
IE2	0.7820	0.7761	0.7892
System	0.6780	0.6718	0.6842

Table 2.3: Results simple continuous time simulation (1000 replications) for example system

allow that our interval width deviates at most one percent of our approximation. We use the sequential method (Law, 2007) to determine the required number of replications. We need 1000 replications. For each replication, we draw a random number [0,1] from the Uniform [0,1] distribution for each component. If this random number is lower than the failure probability, we consider this component as working; otherwise a failure occurred during this time interval. If a failure occurred during the previous time interval, we draw a second uniform distributed random number [0,1]. If this random number is higher than $e^{-\mu\Delta t}$ with μ is 1 divided by the mean time to repair, we consider this segment as been repaired. Once we know the status of the components, we can compute the status of the system. The system availability is the average of the system status of all replications. Furthermore, we compute the 95% confidence interval. Table 2.3 shows the availability and interval of the (sub)system(s).

2.3 Component importance

It is clear that some components have more impact on system availability than others (Rausand & Hoyland, 2004). In this section we discuss a method to measure the component importance. The importance of a component is related to the function(s) it has within the system. A component that is used in two system functions could be highly important for one function, but less important for the other function. There are several ways to measure component importance. Since this research focusses on the design phase of a system, we look for a method to identify bottlenecks or weak points of the system. Once the weak elements of the system are identified, system designers can decide whether improvement of these elements is desired. To identify the weak components within a system we calculate the improvement potential of each component. This variable states how much the availability of the system will increase if the component is replaced by a perfect one (Rausand & Hoyland, 2004). With a perfect element we mean a element that cannot fail. Hence, the component is always available. Consider a system with multiple components. We denote the probability of a working component i at time t by $p_i(t)$ and the probability of a working system by $h(\boldsymbol{p}(t))$. Here, $\boldsymbol{p}(t)$ is the vector of the individual components' probability. The improvement potential of an element i at time t is given by (Rausand & Hoyland, 2004):

$$I^{IP}(i|t) = h(1_i, \boldsymbol{p}(t)) - h(\boldsymbol{p}(t)) \qquad \forall i = 1, \ldots, n \tag{2.19}$$

In practice, it might not be possible to replace a component with a perfect one. We can then use the same equation, but we use the highest possible failure probability that is still realistic. The obtained potential is called credible improvement potential (Rausand & Hoyland, 2004).

Once the improvement potential of all the components are computed, we can rank them to identify the bottlenecks of the system with respect to availability. Availability improvement of the component with the highest improvement potential will result in the largest increase in system availability (Zhang, 2014). Furthermore, we can check whether there are differences between availabilities of subsystems. Unfortunately, the importance of a component does not provide information on how the element could be improved.

Consider again the example shown by Figure 2.3. Suppose the system performs only one function. If

Component	A_1	A_{IE1}	A_{IE2}	A_s	Improvement potential
1	1.0	0.96	0.784	0.7526	0.0753
2, 3	0.9	1.0	0.784	0.7056	0.0282
4, 5, 6	0.9	0.96	0.91	0.7862	0.1089

Table 2.4: Improvement potentials of example system

we replace component 1 by a perfect component, the unavailability of component 1 becomes zero, but the rest of the system remains the same. The new unreliability is

$$Q_0(t) = 1 - \prod_{i=1}^{3}(1 - q_i(t)) = 1 - (1-0)(1-0.04)(1-0.216) = 0.2766. \tag{2.20}$$

$$I^{IP}(i|t) = h(1_1, \boldsymbol{p}(t)) - h(\boldsymbol{p}(t)) = (1 - 0.2766) - (1 - 0.3266) = 0.0753. \tag{2.21}$$

We can calculate the improvement potential of the other components in the same way. Table 2.4 shows the improvement potential of each component. As can be seen, component 4, 5 and 6 have the highest improvement potential. If we want to change one component, it would be one of these. This will result in the largest increase of system availability. This is no surprise, because the availability of these components is relatively low and at least two of them need to work. In general, component improvement costs money. Therefore, we have to make a trade-off between availability improvement and costs.

2.4 Summary

In this chapter we discussed the following points:

- System availability is determined by **component availability** and the **structure of the system**.
- We have discussed five availability calculation models. With **fault trees** and **reliability block diagrams** we model the system by decomposing it into building blocks. With formulas for serial, parallel and k-out-of-n structures we can calculate system availability. For simple systems we can calculate this exactly; for more complex systems we need approximations. **Markov models** consider the system as a whole and its possible states. It can be used if the failure and repair times are exponentially distributed. This method is only suitable for small systems, because the number of system states rapidly increases for larger systems.
  If the system is too large or too complex to calculate (or accurately approximate) the system availability, we can use simulation techniques. **Simple continuous time simulation** makes use of random numbers to simulate the state of each component. These statuses are used to obtain the system status. The simulation is replicated many times. The system availability is the average system status. If we need more details, we can simulate the system in more detail using **discrete event simulation**.
- The **improvement potential** of a component shows how much the availability of the system will increase if the component is replaced by a perfect one. If we want to increase the system availability we will increase the component with the largest improvement potential or the one with the highest improvement per invested euro.

Chapter 3

Current situation

In this chapter we analyse the current situation regarding availability calculation and measurement. By doing so, we answer our second research question. We start with a discussion of the usage of availability calculations within Vanderlande in Section 3.1 and we briefly explain the design process in Section 3.2 and the two availability related industry standards used by Vanderlande. Next, we describe the three availability calculation methods that Vanderlande uses in Section 3.4. Then, in Section 3.5 we describe how availability is measured for operational systems. We end this chapter with a discussion of the strengths and weaknesses of the current availability calculation methods in Section 3.6.

3.1 Usage of availability calculations

Availability calculations are made based on customer's request. However, customers are often satisfied with an availability figure without calculation. As a result, Vanderlande has done availability studies for only a small fraction of their projects. Consequently, there is currently little knowledge on availability and failure behaviour of the system during sale and design phase. This is not necessarily a problem. However, the agreed availability can be defined in a service contract which is requested more and more by the customers. It happens regularly that an agreed availability figure is higher than the measured availability once it is operational at the customer.

3.2 Sales and design process

The sales phase starts with a new lead. The sales manager dives into the requirements of the customer and determines what equipment suits these deliverables. Next, a sales layout is created and the sales manager checks whether or not the project is in line with the long-term goals of Vanderlande. In fact, the project needs to be sold internally. If Vanderlande decides to engage a sales team is formed. This team performs a feasibility and concept study. This includes the generation of a process flow diagram and a material flow diagram.The sales team advises the management of Vanderlande to bid or to quit the project. If Vanderlande wants to bid the customer's requirements are worked out into a quotation. A system concept is made together with a more detailed layout. Furthermore, the price is computed. The quotation is tested on risks such as technical feasibility, price on margin level and legal aspects. If the customer accepts the quotation, the negotiations with the customer start. If the contract is signed, the project is handed over from the sales department to operations.

3.3 Industry standards: FEM & VDI

Within the automated material handling industry there are two industry standards referenced regularly. Since these standards are important in the eye of Vanderlande and its customers, we discuss them here as well. The first standard is called FEM 9.222 and is developed by the European federation of handling (FEM). The standard gives a definition of availability together with formulas how to calculate availability (European Federation of Materials Handling, 1989). FEM implicitly use the fault tree method and the formulas for a serial and parallel structure are the same as discussed in Chapter 2. However, they do not mention how to calculate the availability of a k-out-of-n structure. The other standard is called VDI 3649 and is developed by the association of German engineers (VDI). The standard gives a definition for availability (Verein Deutscher Ingenieure, n.d.). Like the FEM standard they implicitly assume a fault tree decomposition of the system. Again the formulas for a serial and parallel structure are the same as discussed in Chapter 2. Furthermore, they give an example of k-out-of-n system. Although they give some formulas to calculate the availability of the example system, they do not provide generalized formulas that do apply for general k-out-of-n structures.

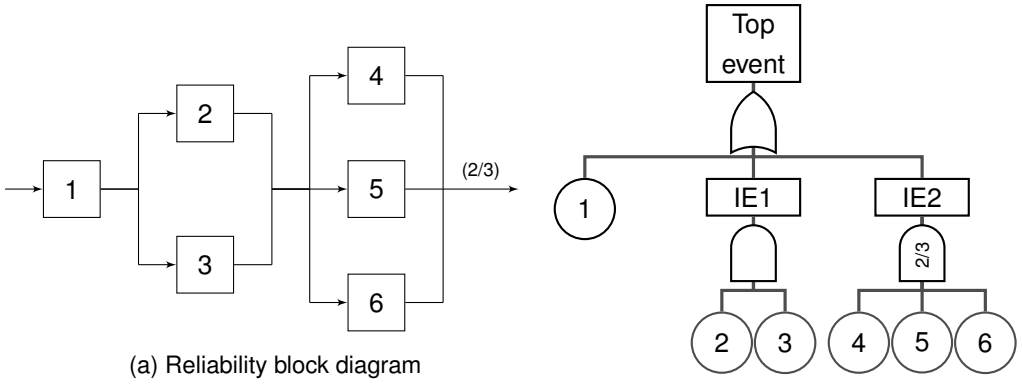
3.4 Vanderlande's availability calculation

Vanderlande uses fault tree analysis (FTA) to calculate the technical availability of designed systems, regardless of their corresponding market segment. The failure tree analysis method is already explained in Section 2.2.1. The lowest level that Vanderlande considers is the equipment level. Examples of components on this level are normal and curved conveyor belts, sorters, and screening machines. The research and development department is concerned with providing the input data. For each component on equipment level, they compute the mean time between failure (MTBF) and the mean downtime (MDT). The department obtains the data by doing a short time study or retrieves the data from the equipment supplier. Below we explain the three calculation methods that are used by Vanderlande. The system's layout and the material flow diagram are used to make the fault tree. This is done manually. Next, the fault tree structure and its availabilities are put in Excel. Formulas for serial and parallel structures are added in order to calculate the availability of the system. These formulas are based on the industry standards FEM 9.222 and VDI 3649. Below we explain the three methods that are used to calculate the availability within Vanderlande.

3.4.1 Method 1

This method is mainly used for smaller baggage handling projects. However, it is sometimes used for other types of projects as well. As described above, the availability calculation starts with the construction of a failure tree. For the first method, this tree is static using serial, parallel or k-out-of-n (k components need to work to have a working system) structures. Figure 3.1 shows an example of a system with its fault tree. As can be seen the system consists of a combination of serial and parallel structures. Of course, component 1 needs to work, but only one of the two elements 2 and 3 needs to work and two out of three of the components 4, 5 and 6 need to work to have a working system.

For this method, instead of calculating the availability of the (sub)system, its complement unavailability is calculated. The following formulas are used to calculate the unavailability of a serial, parallel structure and k-out-of-n respectively:



(a) Reliability block diagram

(b) Fault tree

Figure 3.1: Example of a system consisting of serial and parallel structures

Component i	Availability
1	0.9
2, 3	0.8
4, 5, 6	0.7

Table 3.1: Component availability of example system

$$\bar{A}_s \approx 1 - \prod_{i=1}^{n} (1 - \bar{A}_i) \tag{3.1}$$

$$\bar{A}_s \approx \prod_{i=1}^{n} \bar{A}_i \tag{3.2}$$

$$\bar{A}_s \approx \binom{n}{n-k+1} \bar{A}_i{}^{n-k+1}, \tag{3.3}$$

where $\bar{A}_s$ represents the unavailability of the (sub)system and $\bar{A}_i$ the unavailability of the underlying component i. For a k-out-of-n structure $n-k+1$ components need to fail to have a failed (sub)system. The formulas for a serial or parallel structure are similar to the formulas discussed in Chapter 2. However, the formula for a k out of n structure is different. Moreover, the given formula in (3.3) is not mathematically correct since it is possible to get an (un)availability larger than 1. For example, consider a system where 18 out of 20 identical components need to work to have a working system and suppose the availability of each component is 0.9. Then the unavailability $\bar{A}_s \approx \binom{20}{20-18+1} \bar{A}_i{}^{20-18+1} = \binom{20}{3} 0.1^3 \approx 1.14$ which cannot be true. However, Vanderlande uses this formula because it is a simple approximation. It aims at calculating the probability of the minimal number of failed elements such that the system is down. They assume that the probability that more than this minimal number of elements fail is negligible. We discuss this issue in more detail in Section 3.6.

To come back to the example showed by Figure 3.1, suppose the following component availabilities as showed by Table 3.1.

The unavailability of the two parallel structures and the total system can be computed using formulas (3.1), (3.2) and (3.3):

Component i	Capacity
1, 2, 3	120
4, 5, 6	60

Table 3.2: Capacity of example system

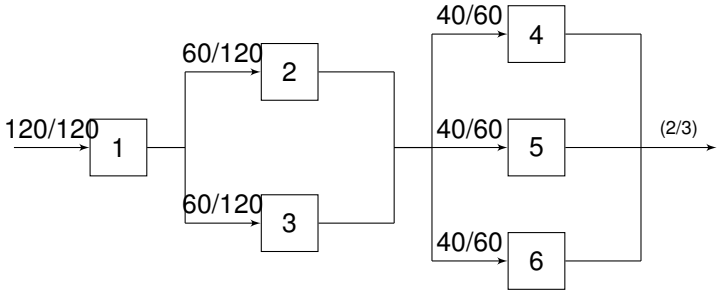


Figure 3.2: Example with flow through the components

$$\bar{A}_{IE2} \approx \binom{3}{2}\bar{A}_4^2 = 0.3^2 = 0.0810 \tag{3.4}$$

$$\bar{A}_{IE1} \approx \binom{2}{2}\bar{A}_2^2 \approx \binom{2}{2}0.2^2 = 0.0400 \tag{3.5}$$

$$\begin{aligned}\bar{A}_{TE} &= 1-(1-\bar{A}_1)(1-\bar{A}_{IE1})(1-\bar{A}_{IE2}) \\ &= 1-0.9\cdot 0.96\cdot 0.919 \approx 0.2060\end{aligned} \tag{3.6}$$

$$A_S = 1-\bar{A}_{TE} \approx 0.7940, \tag{3.7}$$

where the availability of the system is given by A_S, $\bar{A}_{TE}$ stands for the unavailability of the top event and $\bar{A}_{IE1}$ and $\bar{A}_{IE2}$ represents the unavailability of the intermediate events.

3.4.2 Method 2

The second method is the most simple one of the three methods that Vanderlande uses. The method is called the SPA (serial parallel availability). It was originally developed for large baggage handling systems, but is used for other projects as well. The calculation of a series structure is the same as for the previous method.

This method calculates the availability instead of the unavailability. Furthermore, the calculation of the previous method does not take into account capacities of the elements whereas the second method does take this into account. Consider again the example shown by Figure 3.1 and suppose the required throughput is 120 units per hour. Furthermore, suppose the elements have capacities as shown by **??**.

We assume that the flow through the parallel elements is equally spread over the elements. Figure 3.2 shows the block diagram with the corresponding flow through the elements. Since the capacity of elements 2 and 3 is 120, only one of the two elements need to work which was the case in the previous example as well. Furthermore, two elements of the elements 4, 5 and 6 can handle the required throughput, so two out of three elements need to work. This is again in line with the previous example.

Since capacity is taken into account, the formula for parallel strcutures is different from the previous calculation approach. Here, the availability of a parallel structure is the capacity-based weighted average of the availability of the parallel components. The following formulas are used to calculate the availability of a serial and parallel structure:

$$A_s \approx \prod_{i=1}^{n} A_i \tag{3.8}$$

$$A_s = \sum_{i=1}^{n} w_i A_i \tag{3.9}$$

$$\text{with } w_i = \frac{C_i}{\sum_{j=1}^{n} C_j},$$

where w_i is the weight factor for element i which is the capacity of element i divided by the total capacity for the parallel structure. As can be seen, it is simple to calculate the availability of a parallel structure. One can imagine that simple formulas are necessary to obtain availabilities of complex systems in a short period of time. Moreover, during the sales phase the design of the system can change multiple times. To recalculate te availability in a fast way, a simple calculation method is desired. We discuss the consequences of this approximation in Section 3.6.

Again consider the system example depicted by Figure 3.2. The availability of the second parallel structure is equal to the availability of element 4, 5 and 6 since these elements have equal capacity and availability. The same reasoning applies for the first parallel structure. The system's availability can be calculated as follows:

$$A_S = A_1 A_{IE1} A_{IE2} = 0.9 \cdot 0.8 \cdot 0.7 = 0.5040 \tag{3.10}$$

$$\text{with } A_{IE1} = w_2 A_2 + w_3 A_3 = 2 \cdot \frac{120}{240} 0.8 = 0.8$$

$$\text{and } A_{IE2} = w_4 A_4 + w_5 A_5 + w_6 A_6 = 3 \cdot \frac{60}{180} 0.7 = 0.7.$$

3.4.3 Method 3

The last method is the most complex method and is mainly used for parcel & postal projects. As for the previous method, this method takes capacity into account though the calculation for a parallel structure is different. Since parallel elements are often not used at their maximum capacity there is spare capacity which can be used when one or more parallel elements fail. From this perspective, there is no clear k-out-of-n structure, but we rather look at the capacity loss when an element is unavailable which might be only a small fraction of the required throughput.

The following formulas are used to calculate the unavailability of a serial and parallel structure:

$$A_s \approx \prod_{i=1}^{n} A_i \tag{3.11}$$

$$A_s = 1 - \sum_{\forall \omega \in \Omega} \bar{A}_\omega L_\omega \tag{3.12}$$

$$\text{with } \bar{A}_\omega = \prod_{\forall i \in \omega} \bar{A}_i \prod_{\forall j \notin \omega} A_j$$

$$\text{and } L_\omega = \max\{0; 1 - \sum_{\forall i \notin \omega} p_i\},$$

where Ω is the set of all possible combinations of parallel components, $\bar{A}_\omega$ the unavailability of the combination $\omega \in \Omega$, L_ω represents the loss of capacity due to unavailability of ω and p_i is the fraction of required flow that element i can handle.

Consider the example of Figure 3.2. If both component 2 and 3 are working each component handles a flow of 60. However, they both have a capacity of 120. Their utilization is thus 50%. Suppose that

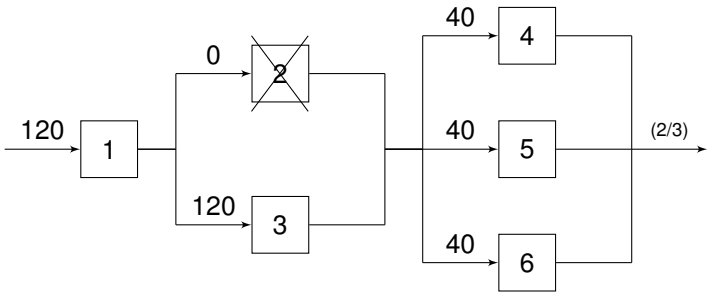


Figure 3.3: Example with flow through the components - component 2 is down

component 2 is down. Figure 3.3 shows this situation. Since component 3 has a capacity of 120 the system can still handle the required flow of 120 and the parallel structure is still available. Only if component 2 and 3 are both down the system will be down. Thus, method 3 takes into account the consequences of lost capacity due to the failure of components. For each combination of working and failed components ω we calculate the loss of capacity L_ω and the unavailability of this combination. The unavailability of the parallel structure is then the sum of the unavailability multiplied by its loss of capacity.

For parallel structures (only one of all components need to work) formula 3.12 is equal to Rausand's method of parallel structures. However, for k-out-of-n-structures it overestimates the availability. We discuss this more detailed in section 3.6.

We return to the example system depicted in Figure 3.2. To calculate the availability of intermediate event 2 (IE2), we first need to list all possible combinations of parallel components and the fraction of flow that each element can handle:

$$\Omega_{IE2} = \big(\{\oslash\}, \{4\}, \{5\}, \{6\}, \{4,5\}, \{4,6\}, \{5,6\}, \{4,5,6\}\big) \tag{3.13}$$

$$p_4 = p_5 = p_6 = \frac{60}{120} = 0.5. \tag{3.14}$$

Next, we calculate the unavailability of each combination and its corresponding loss of capacity:

$$\bar{A}_{\{\oslash\}} = 0.7^3 = 0.3430 \tag{3.15}$$

$$\bar{A}_{\{i\}} = 0.3 \cdot 0.7^2 = 0.1470 \qquad i = 4,5,6 \tag{3.16}$$

$$\bar{A}_{\{i,j\}} = 0.3^2 \cdot 0.7 = 0.0630 \qquad \{i,j\} = \{4,5\}, \{4,6\}, \{5,6\} \tag{3.17}$$

$$\bar{A}_{\{4,5,6\}} = 0.3^3 = 0.0270 \tag{3.18}$$

$$L_{\{\oslash\}} = max\{0; 1 - 0.5 - 0.5 - 0.5\} = 0 \tag{3.19}$$

$$L_{\{i\}} = \max\{0; 1 - 0.5 - 0.5\} = 0 \qquad i = 4,5,6 \tag{3.20}$$

$$L_{\{i,j\}} = \max\{0; 1 - 0.5\} = 0.5 \qquad \{i,j\} = \{4,5\}, \{4,6\}, \{5,6\} \tag{3.21}$$

$$L_{\{4,5,6\}} = \max\{0; 1\} = 1. \tag{3.22}$$

Now we can compute the availability of the parallel structure IE2:

$$\begin{aligned} A_{IE2} &= 1 - \bar{A}_{\{\oslash\}} L_{\{\oslash\}} - 3\bar{A}_{\{4\}} L_{\{4\}} - 3\bar{A}_{\{4,5\}} L_{\{4,5\}} - \bar{A}_{\{4,5,6\}} L_{\{4,5,6\}} \\ &= 1 - 0.3430 \cdot 0 - 3 \cdot 0.1470 \cdot 0 - 3 \cdot 0.0630 \cdot 0.5 - 0.0270 \cdot 1 \approx 0.8785. \end{aligned} \tag{3.23}$$

The same approach can be applied to the parallel structure with elements 2 and 3 (IE1):

$$\Omega_{IE1} = \big(\{\oslash\}, \{2\}, \{3\}, \{2,3\}\big) \tag{3.24}$$

$$p_i = \frac{120}{120} = 1.0 \qquad i = 2,3 \tag{3.25}$$

$$\bar{A}_{\{\oslash\}} = 0.8^2 = 0.6400 \tag{3.26}$$

$$\bar{A}_{\{i\}} = 0.2 \cdot 0.8 = 0.1600 \qquad i = 2,3 \tag{3.27}$$

$$\bar{A}_{\{2,3\}} = 0.2^2 = 0.0400 \tag{3.28}$$

$$L_{\{\oslash\}} = max\{0; 1-1-1\} = 0 \tag{3.29}$$

$$L_{\{i\}} = \max\{0; 1-1\} = 0 \qquad i = 2,3 \tag{3.30}$$

$$L_{\{2,3\}} = \max\{0; 1\} = 1 \tag{3.31}$$

$$\begin{aligned} A_{IE1} &= 1 - \bar{A}_{\{\oslash\}} L_{\{\oslash\}} - 2\bar{A}_{\{2\}} L_{\{2\}} - \bar{A}_{\{2,3\}} L_{\{2,3\}} \\ &= 1 - .6400 \cdot 0 - 2 \cdot 0.1600 \cdot 0 - 0.0400 \cdot 1 = 0.9600. \end{aligned} \tag{3.32}$$

Then, the availability of the system is given by:

$$A_S = A_1 A_{IE1} A_{IE2} = 0.9 \cdot 0.96 \cdot 0.8785 \approx 0.7590. \tag{3.33}$$

This system availability is lower than the availability using the first methods while the system is exactly the same. Furthermore, this method is more cumbersome than the previous methods. Moreover, the number of possible combinations of parallel elements can be very large a structure with more parallel components. We discuss this issue in more detail in Section 3.6.

Before we discuss the shortcomings of the used methods, we first describe the assumptions and the measurement method of Vanderlande.

3.4.4 Assumptions for availability estimation

Vanderlande assumes typically that equipment runs for 20 hours a day. Availability of screening machines is not taken into account, because Vanderlande states that these machines are not part of the scope of supply. Failures of scanners or manual coding station equipment, like computers, are not taken into account in the availability calculations, because Vanderlande states that failure of this equipment will not hamper the baggage flow. Besides these assumptions, Vanderlande has a list of exclusions which can be different between projects. There are operational assumptions such as the exclusion of malfunctions due to material clogging not caused by Vanderlande's (sub)systems or equipment and failures due to products that are outside allowed parameters including weight and dimensions. Furthermore, interruptions or delays caused by operators or unauthorised persons are excluded as well. There could be reasons to exclude non-operational things as well. For example, a common exclusion of Vanderlande is to exclude incipient failures that are detected and repaired without affecting the normal functioning of the system. Sometimes, downtime less than one minute is excluded, because Vanderlande states that such short downtime periods can be neglected in practice. Furthermore, availability of servers and associated controls is often not taken into account because Vanderlande states that their availability is extremely high compared to the availability of the other equipment and will have no significant effect on overall availability of a subsystem.

3.5 Availability measurement

In the last year, Vanderlande developped a new method to measure availability of installed systems. This method defines a material handling system as a series of individual machines (function blocks)

that are placed in a serial or parallel configuration (Lassche, 2015). Larger systems can first be split up into areas that consist of one or multiple function blocks. The method states that this implies certain independency and therefore areas are connected in parallel. Function blocks are defined as parts of the system that serve the same function (Lassche, 2015). If one of the elements fail, the entire function block will fail. Therefore, a function block can be seen as serial elements. A failure can have a technical or operational cause. For each error, the probability that it is a technical or operational error is defined beforehand. If an error occurred causing a certain downtime, this downtime is assumed to be technical downtime for a fraction equal to the technical error probability. The remaining downtime is assumed to be operational downtime. Within a function block it can happen that failures overlap with each other. For example, a second failure occurs while the first failure is not yet repaired. During the overlapping errors, it might not be exactly clear if it is due to a technical or operational cause. Therefore, they use a weighted average to define which part is a technical downtime. Furthermore, minimum and maximum repair times are defined to compensate for operator caused delay repair time. Minimum repair time compensates for start-up time such as travel distance to the failed component. Maximum repair times are used for repair times that are longer than reasonable. For example, repair times might be unreasonably long if the operators are poorly trained. The formulas for serial and parallel structures are the same as the formulas of Method 3. However, they have added the possibility to add relative weights to the function blocks. This is done because some function blocks handle processes that are more critical than others (Lassche, 2015). Since this method is relatively new, there are no projects for which this method is implemented and an availability study was made using method 3.

3.6 Shortcomings of the current situation

In this section we explain the shortcomings of the three used methods for availability calculation within Vanderlande.

The measurement of availability, i.e. the availability calculation once the system is implemented, is different for each of the three calculation methods. This makes it difficult to compare the expected availability with the realized availability. Furthermore, all these methods indirectly assume that all components are independent of each other. However, it is questionable if this is a realistic assumption. If the failure probability is dependent on the load of the component, then components in a parallel structure are dependent on each other. If one of the components fails, the load of the other components increases and hence the failure probability increases. In addition, there are two drawbacks of using fault trees. First, employees can interpret the system structure differently and translate the layout into different fault trees. This results in different availability calculations. Furthermore, manually translating the system into a fault tree and calculating the availability by hand (with the use of spreadsheets) takes a lot of time. Moreover, the quality of the fault tree depends on the time that is available for the availability study. Next to that, none of the methods allows for multiple input types. For example, a customer wanting a new baggage handling system for their airport might require a specific througput for international and domestic check-in baggage and transfer baggage. The methods used by Vanderlande do not provide information how to calculate system availability with these three different input flows. Besides, Vanderlande states many assumptions and exclusions for their availability calculation. It is questionable if the calculation is still realistic, useful and reliable. Finally, the input data is outdated. Many of the component availability figures are not updated for years and it is not precisely known how these figures are obtained.

To summarize, we discussed the following general aspects:

1. Availability calculation during operationalization different from design phase calculation;
2. Independent components assumption;
3. Different employees might construct a fault tree differently;

4. Fault tree construction takes a long time;
5. Methods do not allow for multiple input types;
6. Many assumptions and exclusions;
7. Outdated input data.

Besides the general drawbacks of the methods, there are some drawbacks of each specific method which we will discuss below.

3.6.1 Method 1

This method does not take into account the capacities of the components. This can be a problem when capacities are not identical, because then it is difficult for parallel and k out of n structures to determine how many components need to work. Furthermore, as we have seen already in Section 3.4, using method 1 it is possible to get an (un)availability larger than 1. This is caused by the formula used for k out of n structures, see Equation 3.34. First, this formula assumes that all components are equal, because there is no summation or product over index i. Second, this formula ignores the probability that more than $n-k+1$ components are not working. Vanderlande ignored this probability, because this probability is generally very small. However, the probability that $n-k+1$ identical components are not working is not correctly calculated. Equation 3.35 shows the correct formula. As can be seen, Vanderlande ignores the last part of the equation. This can results in (un)availabilities larger than 1.

$$\bar{A}_s \approx \binom{n}{n-k+1} \bar{A}_i{}^{n-k+1}, \tag{3.34}$$

$$P(n-k+1 \text{not working}) = \binom{n}{n-k+1} \bar{A}^{n-k+1} A^{k-1}. \tag{3.35}$$

3.6.2 Method 2

In contrast to the first method, method 2 is mathematically correct. However, there are issues with the formula for parallel structures (see Equation 3.36). System availability is considered as a capacity based weighted average of the component availabilities. This can be seen as the fraction of total capacity that is available. However, this ignores the required throughput of the structure. Generally, only a fraction of the total capacity of the structure is needed to reach the required throughput. Furthermore, this clearly underestimates the availability of a parallel structure. The power of a parallel structure is that only one (or k in case of a k out of n structure) component needs to work.

$$A_s = \sum_{i=1}^{n} w_i A_i \tag{3.36}$$

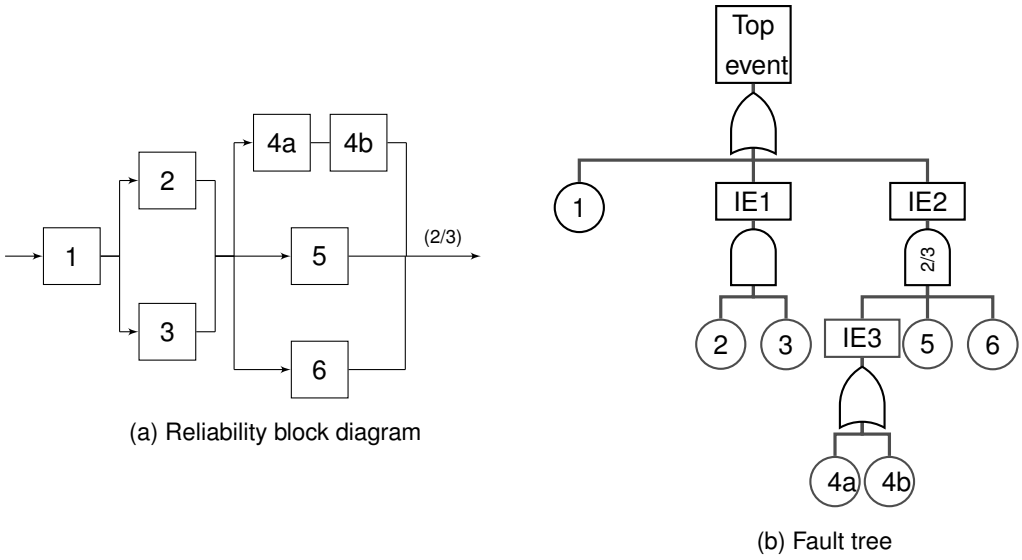
$$\text{with } w_i = \frac{C_i}{\sum_{j=1}^{n} C_j}.$$

3.6.3 Method 3

Regarding method 3, the formula for parallel structures (only one of all components need to work) is equal to the exact method for parallel structures. The fraction of required flow p_i that combination i can handle is either one or zero in this case. Only the combination consisting of all components results in a loss in capacity; for all other combinations the required flow can be handled. However, for k-out-of-n-structures it overestimates the availability. This method allows the system to be partially available.

A_i	Method 1	Method 2	Method 3	Exact method
A_S	0.7940	0.5040	0.7590	0.6774
A_{IE1}	0.9600	0.8000	0.9600	0.9600
A_{IE2}	0.9190	0.7000	0.8785	0.7840

Table 3.3: Availability of example system



(a) Reliability block diagram

(b) Fault tree

Figure 3.4: Extended example system

However, according to our definition of availability defined in Chapter 1, the system will be unavailable if the required throughput cannot be met. Furthermore, the difference in k-out-of-n structure's availability between method 3 and the exact approach can become larger when the component availabilities are not identical.

3.6.4 Comparison

Table 3.3 shows the availabilities of the (sub)systems for each method, together with the exact calculation, which we discussed in the Chapter 2. For this small example, we can clearly see differences between the computed availabilities. For large and complex systems such as the systems that Vanderlande designs, the differences will be even larger. We can see that the availability of the parallel structure (IE1) of method 1 and 3 are in line with the exact calculation. Method 2 underestimates parallel structures, resulting in a lower availability for structure IE1. Method 2 underestimates the availability of the k out of n structure (IE2) as well. In contrast, method 1 and 3 overestimate the availability of structure IE2. This results in an overestimation of the system availability as well. Method 2 gives an underestimation of the system availability.

Consider again our example system, depicted in Figure 3.1a, but now component 4 is split into to components, 4a and 4b. The two components are in series and suppose that both components have availability of 0.7. Figure 3.4 shows this new situation. Table 3.4 shows the obtained availabilities of each method. As we can see, method 2 (3) still underestimates (overestimates), but now relatively more. For method 1, we cannot calculate the availability of the k out of n structure, because element 5 and 6 have an availability of 0.7, whereas component 4a and 4b together have an availability of $0.7^2 = 0.49$. Since the formula for k out of n structure assumes identical components, we cannot evaluate the availability of this structure.

We return to our first example system, with only one component 4. Suppose now that the components (and their availabilities) are identical. Figure 3.5 shows the system availability for varying component availability.

A_i	Method 1	Method 2	Method 3	Exact method
A_S	?	0.4536	0.7128	0.6012
A_{IE1}	0.9600	0.8000	0.9600	0.9600
A_{IE2}	?	0.6300	0.8250	0.6958

Table 3.4: Availability of example 2 system

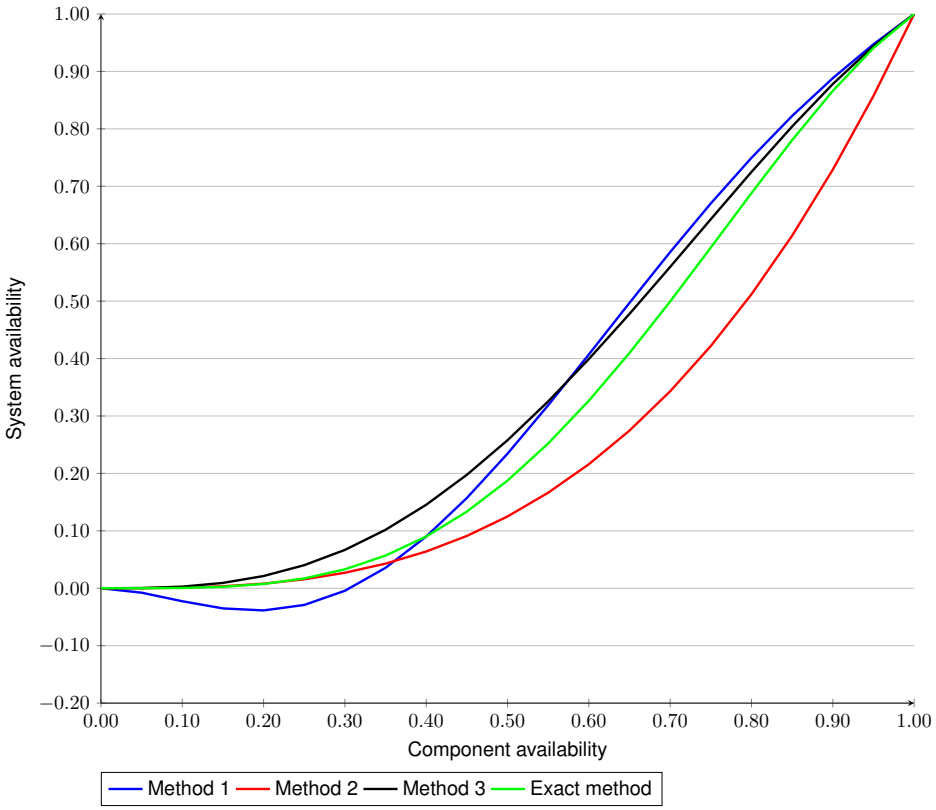


Figure 3.5: System availability of example system

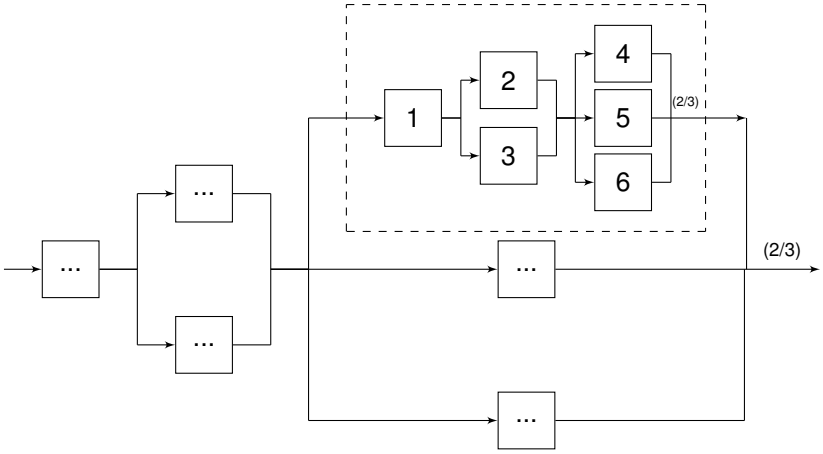


Figure 3.6: Example system with $6^2 = 36$ components

We can see that method 1 results in negative system availability for small component availabilities. Furthermore, method 2 underestimates the system availability regardless the availability of the component. In contrast, method 3 overestimates system availability. Method 2 overestimates for component availabilities larger than 0.40. The graph shows that even for small systems, the differences between the methods are clear. Vanderlande typically has systems with highly reliable components. The difference between the exact calculation and method 1 and 3 becomes smaller as the component availability comes close to 1. However, method 2 gives even for large component availabilities substantially lower system availability than calculated in the exact way.

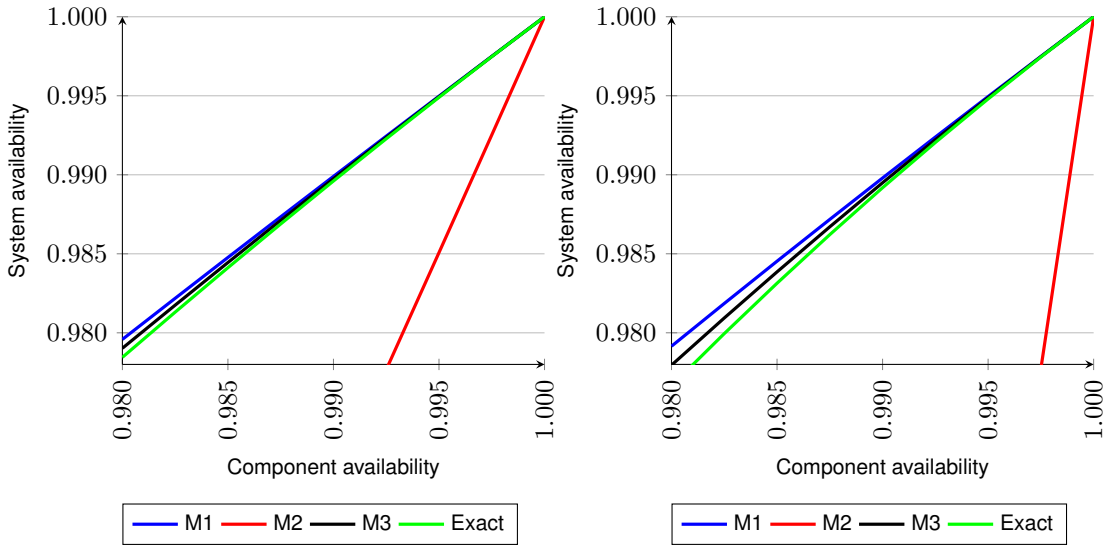
The equipment that Vanderlande uses such as conveyor belts, sorters and screening machines, are generally highly reliable. Therefore, their availability is close to one. Most of the parts have availability above 99.5% and we can state that 98% is the minimal equipment availability. When we zoom in on availabilities close to one, method 1 and method 3 are almost equal to Rausand's exact method. Figure 3.7a) shows this. However, the systems of Vanderlande have typically thousands of components. So suppose that each component of our example system actually consists of a subsystem with the same layout. Figure 3.6 shows this example. As can be seen, this system consists of six subsystems with six components each. Thus, we have $6^2 = 36$ components in this system. Now assume that each component in the subsystems on its turn consists of a subsystem as well. Then there are $6^3 = 216$ components within the system. We can further expand this system into a system with 6^n components. As the number of components within the system increases, the deviation of each of Vanderlande's methods from the exact method becomes larger. Figure 3.7 shows the system availability as a function of the component availability for $n = 1, 2, 4, 6$.

Now we have seen how the current methods perform, we can move to the development of a new availability calculation model. In the next chapter we discuss the requirements for this model and we define four alternative calculation models. But first we summarize our findings regarding the current situation.

3.7 Summary

In this chapter we discussed the following points:

- Currently, Vanderlande has little knowledge on availability and failure behaviour of the system during sale and design phase.
- We discussed the sales and design process, from new lead to signed contract.
- Vanderlande uses three methods to calculate the availability of a system. All methods are based on **fault tree analysis**. Fault tree construction is the same for every method, but the formulas to



(a) $6^1 = 6$ components

(b) $6^2 = 36$ components

(c) $6^4 = 1296$ components

(d) $6^6 = 46656$ components

Figure 3.7: System availability of four example systems varying in size

	Series	Parallel	k-out-of-n
Method 1	$\bar{A}_s \approx 1 - \prod_{i=1}^{n}(1-\bar{A}_i)$	$\bar{A}_s \approx \prod_{i=1}^{n} \bar{A}_i$	$\bar{A}_s \approx \binom{n}{n-k+1} \bar{A}_i^{\,n-k+1}$
Method 2 (SPA)	$A_s \approx \prod_{i=1}^{n} A_i$	$A_s = \sum_{i=1}^{n} w_i A_i$ with $w_i = \frac{C_i}{\sum_{j=1}^{n} C_j}$	see parallel
Method 3	$A_s \approx \prod_{i=1}^{n} A_i$	$A_s = 1 - \sum_{\forall \omega \in \Omega} \bar{A}_\omega L_\omega$ with $\bar{A}_\omega = \prod_{\forall i \in \omega} \bar{A}_i \prod_{\forall j \notin \omega} A_j$ and $L_\omega = \max\{0; 1 - \sum_{\forall i \notin \omega} p_i\}$	see parallel
Exact method	$\bar{A}_s = 1 - \prod_{i=1}^{n}(1-\bar{A}_i)$ for independent components	$\bar{A}_s = \prod_{i=1}^{n} \bar{A}_i$ for independent components	$\bar{A}_s = \sum_{y=k}^{n} \binom{n}{y} \bar{A}_i^{\,y} (1-\bar{A}_i)^{n-y}$ for independent and identical components

Table 3.5: Vanderlande's three calculation methods versus exact method from literature

calculate the system availability are different (see Table 3.5).
Method 1 is mainly used for smaller baggage handling projects whereas method 2 was originally developed for large baggage handling systems. Method 3 is mainly used for parcel & postal projects and warehouse automation projects. For all three methods, Vanderlande has defined **assumptions and exclusions** regarding the availability of the system. These can differ among projects.

- Last year, Vanderlande developed a new method to **measure availability** of installed systems. This method is similar to method 3.
- We identified seven **shortcomings** that hold for each method. Among them is that fault tree construction takes much time and that the models are **static**. Furthermore, we discussed the method-specific disadvantages of each method. We have seen that the formula for k-out-of-n structures of method 1 is mathematically incorrect which can result in negative availabilities. Method 2 underestimates the availability of a parallel structure whereas method 3 overestimates the availability of a k-out-of-n structure. For a small and simple system with highly reliable components, method 1 and method 3 perform well. Method 2 substantially underestimates the availability even for a small and simple system. For larger or more complex systems, such as the systems of Vanderlande, all three methods compute **deviating availabilities** compared to the exact method from literature.

Chapter 4

Model selection

In this chapter we answer our third research question: what is the most appropriate modeling approach for availability calculation for Vanderlande? First we list the requirements of the availability calculation model in Section 4.1. Next, we give four alternative models in Section 4.2.1. We end this chapter with a discussion of the (dis)advantages of each model and a decision for one of the alternatives in Section 4.2.2.

4.1 Model requirements

In agreement with interested and concerned employees of Vanderlande, we define a priority ranked list of requirements which is shown below. This is a difficult task, because there are many different views on what the model should be able to do. Whereas some persons mention some aspects of the model as the most important ones, others mention these as completely unimportant. We asked involved employees to rank potential requirements, provide feedback and propose complementary requirements. We aggregated the results and after a few adjustments, everyone involved agreed on the list of requirements. We make a distinction between high and low priority requirements. The high priority requirements are strict requirements. These requirements must be met. The low priority requirements can be seen as features that are nice to have, but are not strict necessary. We first discuss the high priority requirements followed by the low priority requirements.

4.1.1 High priority requirements

The first requirement is system standardization. With this we mean that the model uses the same method to calculate the availability of a system regardless of the business unit (baggage handling, parcel & postal and warehouse automation) it belongs to. The second requirement is the model should not only able to calculate the availability of a whole system, but it should be able to compute the availability of the subsystems as well. User standardization is the third requirement. By this we mean that if two people use the model to obtain the availability of the same system the model results should be the same as well. The model will be used in the sales and design phase. During this phase, Vanderlande and its customer agree upon the required availability of the system. It is, therefore, important that the model is easy to explain to the customer. Requirement four describes this. The last high priority requirement is that the model uses a mathematically correct method. Recall that method 1 of Vanderlande is not mathematically correct. However, Vanderlande indicated that a mathematically correct method is important. Although this sounds as an obvious requirement, it is important to have this requirement, because not every organization in the industry attaches importance to this. To summarize, we list the high priority requirements below:

1. System-standardization: same method for each business unit
2. Output in terms of system availability as well as subsystem availability
3. User-standardization: same answer regardless of the user
4. Easy to explain the model (and its results) to the customer
5. Mathematically correct method

4.1.2 Low priority requirements

The first low priority requirement is to have a flexible model. By this we mean that the model allows for quickly calculating the change in availability for small layout changes. The second requirement is a model that can be automated in the future. The currently used methods are labour-intensive and it would be nice to automate the calculation process. The third requirement is to have a model that allows for multiple input types. For example, a model that can differentiate between check-in and transfer baggage. A model that is in line with the availability calculation of installed systems is the fourth low priority requirement. We discussed the availability measurement in Chapter 3.5. The last requirement is that the model should be in line with the industry standards. Customers value these standards and therefore it is important for Vanderlande as well. We discussed the industry standards in Chapter 3.3. To summarize we list the low priority requirements below:

1. Flexible model: allows for quickly calculating the change in availability for small layout changes
2. Model that can be automated in the future
3. Model that allows for multiple input types (for example, check-in and transfer baggage)
4. Model that is in line with availability calculation of installed systems
5. In line with industry standards

Together with Vanderlande we discussed more possible requirements than those that are listed above. It turns out that Vanderlande does not value a science-based method. However, since this project is a research project our model will be science-based. Other aspects that are found to be unimportant are, among others, an intuitively correct method, a small required manual calculating time, a small required computation time, low development cost and time and little required knowledge to be able to use the model and low development costs.

4.2 Choice of the model

In this section we start with a description of three alternative models to calculate the system availability of Vanderlande's systems. Next, we discuss the advantages and disadvantages of each alternative and we make a decision which model we will further develop.

4.2.1 Model alternatives

We base our alternatives on the different methods discussed in our theoretical framework (Chapter 2). Reliability block diagrams are similar to fault tree analysis. Therefore, we do not consider this method. Since using Markov models will become too complex due to the state space explosion we do not consider this method either.

Alternative 1: Fault tree analysis

The first alternative is based on Vanderlande's third method. To recapitulate, we first make a fault tree of the system and then calculate the (un)availability of the system using formulas for serial (OR gate) and parallel (AND gate) structures. These formulas assume independent components. Therefore, we have to check if this assumption is realistic and keep in mind that the results are only an approximation.

Vanderlande's third method is not in line with our definition of availability. Therefore, instead of using the loss of capacity due to unavailability (L_ω) we introduce a binary indicator I_ω showing whether or not combination ω can handle the required flow. Furthermore, since our formulas are approximations, we use the 'approximately equal to' sign instead of the equal sign. Equations 4.1 and 4.2 show the new formulas to calculate the availability of a serial or parallel structure:

$$A_s \approx \prod_{i=1}^{n} A_i \tag{4.1}$$

$$A_s \approx 1 - \sum_{\forall \omega \in \Omega} \bar{A}_\omega I_\omega \tag{4.2}$$

$$\text{with } \bar{A}_\omega = \prod_{\forall i \in \omega} \bar{A}_i \prod_{\forall \notin \omega} A_j \tag{4.3}$$

$$\text{and } I_\omega = \begin{cases} 1 & \text{if } \sum_{i \notin \omega} p_i < 1 \\ 0 & \text{otherwise,} \end{cases}$$

where Ω is the set of all possible combinations of parallel components, $\bar{A}_\omega$ the unavailability of the combination $\omega \in \Omega$, I_ω is an binary indicator showing whether or not combination ω can handle the required flow and p_i is the fraction of required flow that element i can handle.

When we apply these formulas to our example system (see Figure 2.5) we get the same results as for the exact method from literature. Although it gives exact results for this example, this will not be the case for all systems due to the approximation formulas. Furthermore, since the availability measurement method is based on Vanderlande's third method, it is easy to align the measurement method with this alternative to provide a proper feedback loop. However, fault tree construction still takes much time and it is difficult to standardize. Moreover, this method does not allow for multiple input types such as check-in and transfer baggage. Furthermore, we have to update the input data to obtain reliable results.

Alternative 2: Simple time simulation

For the second alternative we use a simple time simulation. This allows for modelling dependencies between components. Furthermore, by simulating the system we can obtain a reliable estimate for the availability of the system. Below we briefly describe the simulation method.

1. Define the system input types. A system input type is a required peak throughput of a (group of) material(s) sharing equal characteristics for the system. For a baggage handling system, the system input types might be international check-in baggage, domestic check-in baggage and transfer baggage. If there are multiple system input types, assign strategic weights to them. The transfer baggage might be twice as important as the check-in baggage. Therefore, we give transfer baggage a weight 0.5 and international and domestic check-in baggage each a weight 0.25.

2. Decompose the system into equipment. Examples of equipment are a conveyor belt, a screening machine and an automatic case picking machine. Calculate the failure rate λ_i and repair rate μ_i for each equipment i using the following formula (we assume constant failure and repair rates):

$$\lambda_i = \frac{1}{MTTF_i} \tag{4.4}$$

$$\mu_i = \frac{1}{MTTR_i} \tag{4.5}$$

Furthermore, make sure you know the capacity of the equipment, the weighing factor, the input types it has to handle, to which subsystem it belongs and the position of each equipment in the system. The latter can be done by defining its upstream component(s) and its downstream component(s).

3. Perform a simple time simulation. We discretize the time domain into small intervals of equal size. We assume that the mean time to repair is larger than the mean time to failure.

   (a) For each time interval:

      i. For each equipment that was functioning during the previous time interval, generate a random number [0,1] from the uniform distribution for each segment. If this random number is lower than $e^{-\lambda \Delta t}$, we consider this equipment as working and the equipment state is 'functioning', otherwise we change the equipment state to 'failure'. We assume that all equipment are working at time zero;

      ii. For each equipment that was in failed state during the previous time interval, draw a uniform distributed random number [0,1] as well. If this random number is higher than $e^{-\mu \Delta t}$, we consider this equipment as been repaired and we change the equipment state to 'functioning', otherwise the item is still under repair and its segment state is 'failure';

   (b) Compute the fraction of input flows that comes out of the (sub)system. If an equipment cannot handle the complete throughput, it handles the relative flows according to their weights.

   (c) Calculate for each input flow its availability by taking the weighted average of outputs/inputs over time. The (sub)system availability (of the current replication) is the weighted average of the availability of the input flows.

   (d) Add more replications of the simulation. Repeat steps a) to c) until desired accuracy is obtained.

4. The expected (sub)system availability is the average of the (sub)system availabilities obtained in the replications.

Alternative 3: Detailed time simulation

The third alternative makes use of the simulation models of the simulation department. These models are detailed time simulations. Instead of using abstract input flows, we now simulate the actual handling of the materials. For example, for a baggage handling system, the simulation model graphically shows how the luggage moves through the system. Failure behaviour of the equipment should be modelled as well to compute the expected availability of a system. The main advantage of a detailed simulation is that the simulation is closer to reality resulting in reliable outputs. However, the simulation department does not make a simulation for each Vanderlande's system and modelling the system and its failure behaviour can take a lot of time. Moreover, the execution of the simulation will take a lot of time as well. Finally, debugging the modelled failure behaviour can be hard due to the many variables and dependencies that define the failure behaviour.

4.2.2 Comparison of the three alternatives

Now we have three alternatives, we can weight them and choose the best to develop further. All three alternatives allow for system standardization. Simulation techniques as well as fault tree analysis can be applied in the same way to all business units. This does not hold for user standardization. The construction of a fault tree is not straightforward and a system can be translated into a fault tree in different ways. Of course, we can develop guide lines for the fault tree construction, but it is hard to define them in a way that results in user standardization. In contrast to the first alternative, alternative two is suitable for user standardization. If enough replications of the simulation are made, the results are about the same. For the detailed time simulation it is a bit harder to obtain user standardization. Since simulation engineers need to model many details, it is difficult to ensure that every engineer interprets the system and its details in the same way. On the other hand, a detailed and visual simulation is well explainable to the customer. This is not the case for the simple simulation. Since this is an abstract simulation, it might be difficult to explain how the simulation is done. The customer might get an idea of a black box: data are put in and magically an availability figure is thrown out. For alternative one this is less of a problem. Although the mathematical formulas might be difficult to understand, the idea of the method is easy to explain. Table 4.1 shows the score on the (high and low) requirements for each alternative. With respect to the high priority requirements, alternative 2 scores the best. However, if it is harder to meet the high priority requirements, but an alternative can easily meet the low priority requirements, we might prefer this alternative.

When we look at Table 4.1 we do not see an alternative that scores high on every low priority requirement. While the fault tree analysis method allows for multiple input types, is in line with the availability measurement method and in line with the industry standards, it is not a flexible method. The fault tree method is a static method. Therefore, we might have to locally change the fault tree even for small changes in the system layout. Furthermore, it is difficult to automate the fault tree construction process. Alternative two scores almost opposite compared to the fist alternative. This method is not in line with the industry standards and it takes some effort to get this method and the availability measurement method in line. On the other hand, it is a flexible method that can easily be automated in the future. Like the simple simulation, the detailed time simulation is not in line with the industry standards and availability measurement method. However, detailed simulation is not very flexible and it is difficult to automate the simulation modelling process. On the other hand, detailed time simulation can easily handles multiple input types.

After comparing the alternatives, we decide to use a simple time simulation for our availability calculation model. We are well aware that we have to think carefully about the way how to explain the model to the customer. Nevertheless, this alternative scores best on all the other high priority requirements. Besides, it should be possible to meet some of the low priority requirements as well. In the next chapter we describe the working of our models and the input data that is needed to calculate the expected availability. Furthermore, we explain how we test and validate the model.

4.3 Summary

In this chapter we discussed the following points:

- **High priority requirements** are system standardization, option to have subsystem output, user standardization, easily explainable to the customer, and a mathematically correct method. These high priority requirements must be met.
- **Low priority requirements** are a flexible and automatable model, option to have multiple input types, a model that is in line with Vanderlande's availability measurement method and in line with the industry standards. These low priority requirements are not necessary, but are nice to have.

Requirements	Alternative 1	Alternative 2	Alternative 3
High priority			
1. System standardization	+ +	+ +	+ +
2. Subsystem output	-	+	+
3. User standardization	- -	+	-
4. Easy to explain	+ -	+ -	+
5. Mathematically correct	+ -	+ +	+ +
Low priority			
1. Flexible model	- -	+ +	+ -
2. Automatable in future	+ -	+	+ -
3. Multiple input types	-	+	+ +
4. In line with measurement	+ +	+ -	+ -
5. In line with industry standards	+ +	-	-

Table 4.1: Aggregated requirements scoring for the four model alternatives

- We discussed three alternatives to model the availability of a material handling system: **fault tree analysis method, simple time simulation and detailed time simulation**. We weighted the alternatives against each other and we decided to use a **simple time simulation** to calculate the expected (sub)system availability.

Chapter 5

Model description: simple time simulation model

In this chapter we describe our availability calculation prototype model. We first describe the required input parameters in Section 5.1. Next we describe the model itself in Section 5.2 and in Section 5.3 we describe the software tool we develop for it. In Section 5.4 we discuss the running time of the software model. We give describe an example system in Section 5.5. We end the chapter with Section 5.6 in which we check if requirements are met and in which we discuss the verification of the model.

5.1 Input parameters

We need input data to calculate the expected availability of a material handling system. Vanderlande has, among others, digital layout drawings of the system, a list of equipment and flow diagrams describing the flow throughout the system. However, with this information we currently cannot retrieve the required input data automatically. Due to the time limitation we focus on the more important question on how to use this data. The only option is to use the available data and load it manually into our model. In the rest of this section, we describe consecutively the required equipment data, material data and other parameters that need to be set.

5.1.1 Equipment data

For each equipment we need information about how the equipment behaves and what its relation is with other equipment. Table 5.1 shows all data we need of each equipment. We start with giving each equipment a unique identity number (ID). To be able to calculate the flow through the equipment, we need to know its capacity. The material flow diagram provides this information. The relation with other equipment is defined by its upstream and downstream equipment and can be found by looking at the system's layout. Furthermore, we need to know the average failure and repair behaviour of the equipment. Therefore, we want to know the failure and repair distribution of each equipment. Currently, Vanderlande does not have these distributions. Nevertheless, the research and development department does have the mean time to failure and mean time to repair of each equipment. Although these data might be outdated or not accurate, we use these means. In the future, Vanderlande can collect and process availability data of equipment to obtain more accurate and recent failure and repair time data. Since we do not know the failure and repair time distributions, we assume that each equipment has an exponential failure and repair time distribution. This implies that the failure and repair rate are constant. This assumption is reasonable, since it leads to realistic life time models (Rausand & Hoyland, 2004). It is the most commonly used life time distribution in reliability analysis.

Input	Data source
ID	User
Capacity	Material flow diagram
Upstream equipment' id	Layout
Downstream equipment' id	Layout
Failure distribution	R&D
Mean time to failure	R&D
Repair distribution	R&D
Mean time to repair	R&D

Table 5.1: Input data for each equipment within the system

5.1.2 User's based parameters

The user of the simulation model has to set a number of parameters as well to obtain the availability figures he or she wants. The user can choose a time horizon over which the availability has to be calculated. The time horizon can be set, for example, to a day, week, month, or year. The horizon will be split into small time periods of ten minutes. We discretize time to make the model suitable for numerical evaluation. Furthermore, the user can define how many operational hours go into a day and how many operational days go into a week, month or year. Next to the time horizon, the user has to define its desired level of significance of the calculations. Therefore, the significance level and interval width of the confidence interval has to be set. Section 5.2 provide further information about the significance level of the calculations.

5.2 Availability calculation model

Our model makes use of a simple time simulation as we discussed in Chapter 4. The simulation consists of a series of replications. In each replication we simulate for each time period the state of each equipment. The state of an equipment depends on the previous time period. If an equipment failed during the previous time period, we first need to repair it. On the other hand, if the equipment was functioning during the previous time period, it is more likely that it will work during the current period. We can compute the state of the system and subsystems based on the state of the individual equipment. Once we have simulated the system over the complete time horizon, the availability of the system is simply the number of time periods that the system was functioning divided by the total number of time periods. To obtain reliable results, we do many replications and take the average availability.

In the previous section, we described the input data and parameters for the model. Important parameters are the significance level and width of the confidence interval. The lower the significance level, the more replications are needed to obtain the required significance. The same holds for the width of the interval. The lower the interval width, the more replications we need. We start the simulation with an infinite width of the confidence interval. After each replication we calculate the current confidence interval width. If the interval width is larger than the required input interval width, we add another replication of the simulation. Furthermore, we define an empty array to store the results of all replications. For each replication we first have to do an initialization. We start with a system for which all the equipment are functioning. However, this might not be a realistic starting situation. We then should increase the time horizon by one. We do not use the results of the first period; it is used as warm-up period. After this period the state of the system represents a realistic situation. Next to the warm-up period, we define a counter for each subsystem to keep track of the number of times that the (sub)system is in functioning state over the complete time horizon. We then loop through the time horizon and compute the state of each equipment. If the equipment worked in the previous time period we compute the probability that the equipment will fail in the current time period. Likewise, if the equipment was in failed state in

the previous time period, we compute the probability that the equipment will be repaired in the current time period. As described in Section 5.1.1 Vanderlande has only limited information about the failure and repair behaviour of their equipment. We only know the mean time to failure (MTTF) and mean time to repair (MTTR) of the equipment. Therefore, we assume constant failure rates and repair rates. This implies an exponential failure distribution and exponential repair distribution. The probability that equipment i will fail during a time period Δt given that it is working at the beginning of the period is $1 - e^{\frac{-\Delta t}{MTTF_i}}$. The probability that equipment i will be repaired during a time period Δt given it was on failure at the beginning of the period is $1 - e^{\frac{-\Delta t}{MTTR_i}}$ (Rausand & Hoyland, 2004). In the future, Vanderlande could collect more data about failure and repair behaviour of their equipment to obtain more accurate failure and repair distributions. This will improve the accuracy of the availability calculation. Once we have the (simulated) state of each equipment, we can compute the flow through the system and its subsystems during the current time period. If the flow is larger than the required flow, we consider the (sub)system as available and we increase the counter of the (sub)system, the number of time periods that the (sub)system is available, by one. After we have simulated the system over the complete time horizon, we compute the availability of the (sub)system of the current replication. It is simply the counter, which is the number of times the (sub)system was available, divided by the number of time periods in the time horizon. We store this availability in the array we defined at the beginning of the simulation. The replication is now complete. We compute the new width of the confidence interval (Law, 2007). As described above, if the width of the interval is larger than the required input width, we add another replication. Otherwise, the results are accurate as requested. The availability of the (sub)system is the average availability over all the replications except in the warm-up period. To summarize our approach, we stepwise describe the simulation algorithm below:

1. Initialization: set width of confidence interval to infinity. Define for each (sub)system j an empty array to store the results of all replications;
2. Add an replication:
   (a) Initialization: set each equipment status to working. Set counter, which is the number of times the (sub)system is available, for each subsystem j: $n_j = 0$;
   (b) For each time period:
      i. For each equipment:
         A. If equipment worked in the previous time period, compute the failure probability based on the failure distribution and mean time to failure. Draw a uniform random number between zero and one. If this number is larger than the failure probability, consider the equipment as working. The state of the equipment does not change. Otherwise, change the equipment state to failure.
         B. If the equipment is in failed state in the previous time period, compute the repair probability based on the repair time distribution and mean time to repair. Draw a uniform random number between zero and one. If this number is higher than the repair probability, consider the equipment as been repaired and set its status to working. Otherwise, the equipment has not been repaired and the equipment remains in failure state.
      ii. For each subsystem (including whole system):
         A. Compute the flow through the (sub)system using a mathematical optimization technique (see Section 5.2.1).
         B. If the flow is larger or equal to the required peak flow, consider the (sub)system as available. Set $n_j = n_j + 1$.
   (c) Compute availability of the (sub)system j of the current replication i: $A_{i,j} = \frac{n_j}{T}$, with T the number of time periods.

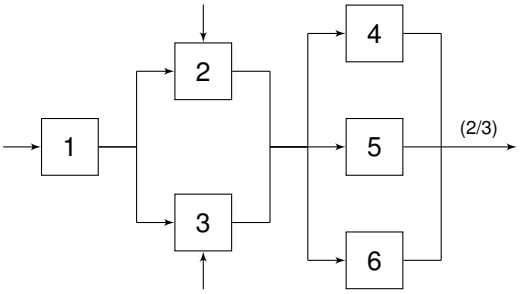


Figure 5.1: Reliability block diagram of extended example system

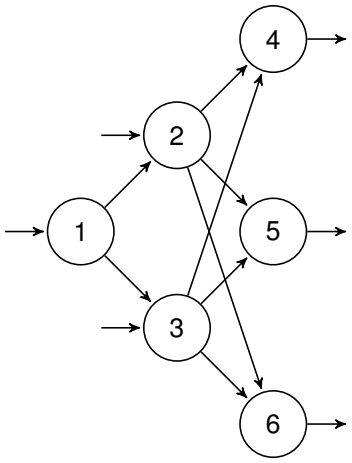


Figure 5.2: Network representation of the example in Figure 5.1

(d) Add $A_{i,j}$ at the end of the results array for (sub)system j.

(e) Compute the width of the confidence interval.

3. If the width of the confidence interval is larger than the required interval width, go to step 2. Otherwise, stop. The availability of the (sub)system is the average availability over all the replications (except the warm-up period).

5.2.1 Flow computation

An important step in the availability calculation algorithm is the computation of the flow through the (sub)system. We use linear programming to compute the maximum flow through the system at a specific time period. Linear programming is a well-known method to solve optimization problems in an efficient way. The method consists of an objective function to be maximized or minimized under the condition that all constraints are met. We refer the reader who is not familiar with linear programming to the book of Winston (Winston, 2003). Our optimization problem falls within the category of network problems. These are problems that can be analysed using a graphical network representation (Winston, 2003), (Manthey, 2015). The systems of Vanderlande can indeed be represented as a network. Nodes represent the equipment of the system. Arcs show the link between the equipment and show the direction of the flow. In Chapter 2 and 3 we used a simple example system. To illustrate the flow computation method, we slightly extended this example by adding more input flows. Figure 5.1 shows this extended example. Figure 5.2 shows the network representation of this system.

We base our linear program on one of the main network flow problems: the flow maximization problem. This problem maximizes the flow from one starting node, the source s, to one end node, the sink t. The problem is formulated as follows:

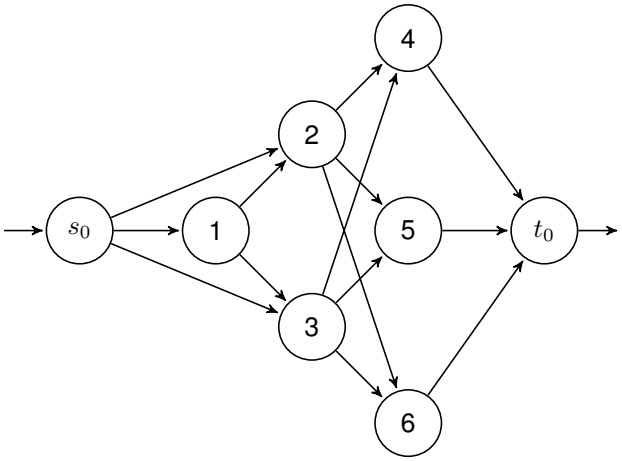


Figure 5.3: Network representation of the example in Figure 5.1 with artificial source and sink

Sets: vertices	$s, u, v, w, t \in V$
edges	$(s, v), (u, v), (v, w) \in E$
Parameters: capacity	$c(u, v)$
Variables: flow	$f(s, v), f(u, v), f(v, w)$

$$\text{maximize} \sum_{v:(s,v)\in E} f(s,v) \tag{5.1}$$

$$\text{subject to} \sum_{u:(u,v)\in E} f(u,v) = \sum_{w:(v,w)\in E} f(v,w) \qquad \forall v \in V - \{s,t\} \tag{5.2}$$

$$f(u,v) \leq c(u,v) \qquad \forall (u,v) \in E \tag{5.3}$$

$$f(u,v) \geq 0 \qquad \forall (u,v) \in E. \tag{5.4}$$

The objective function is to maximize the sum of the flows that depart from the starting node s. The objective function is restricted by three constraints. The first constraint, Equation 5.2, ensures that the incoming flow at a node equals the outgoing flow. This must hold for all nodes except the source s and the sink node t. The second constraint, Equation 5.3, ensures that the flow through an edge does not exceeds the capacity of the edge. The third constraint, Equation 5.4, ensures that the flow through an edge is larger than or equal to zero.

We have to adjust the standard maximum flow problem to fit the flow optimization problem of the systems of Vanderlande. The standard problem takes into account the capacity of the edges whereas we need to ensure that the flow through the nodes does not exceed the node capacity. Furthermore, the systems of Vanderlande could have multiple input and output points. For example, a baggage handling system often has multiple check-in desks and several conveyor loops from which the baggage is loaded into the airplane. Therefore, we define an artificial source node s_0 and sink node t_0. The artificial source node is connected with all real starting nodes. All real ending nodes are connected with the artificial sink node. The artificial source and sink node both have infinite capacity. Figure 5.3 shows this adjustment for the example system. The adjusted maximum flow problem is formulated as follows:

Sets: vertices	$s_0, u, v, w, t_0 \in V$
edges	$(s_0, v), (u, v), (v, w) \in E$
Parameters: capacity	$c(v)$
Variables: flow	$f(s_0, v), f(u, v), f(v, w)$

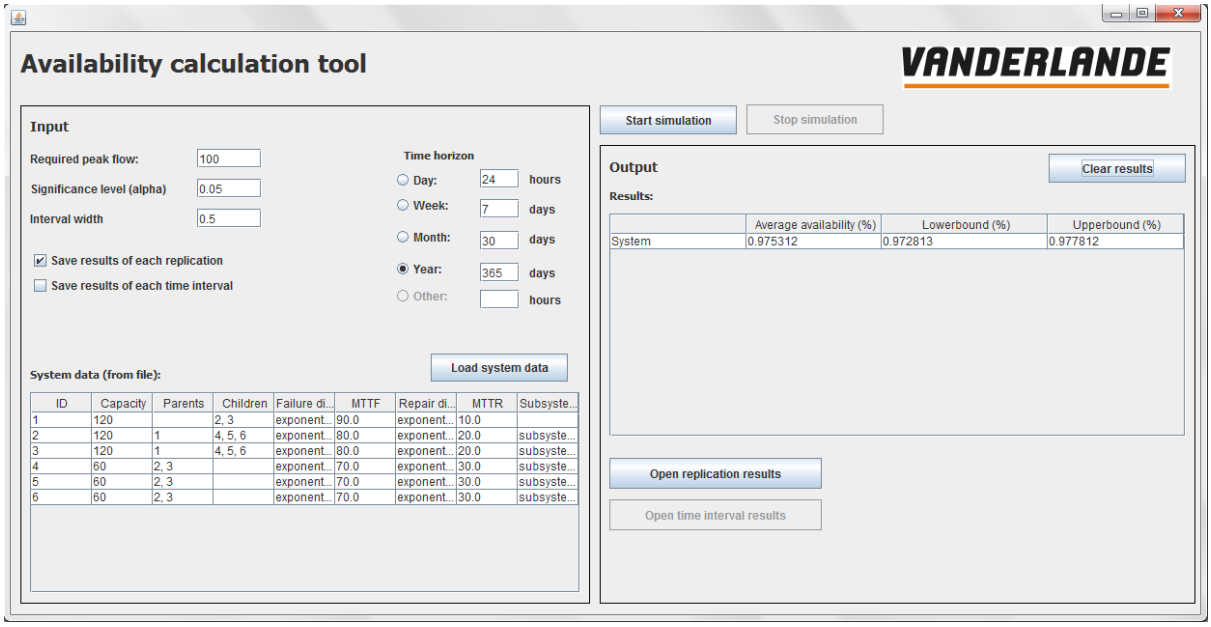


Figure 5.4: Availability calculation tool

$$\text{maximize} \sum_{v:(s_0,v)\in E} f(s_0,v) \tag{5.5}$$

$$\text{subject to} \sum_{u:(u,v)\in E} f(u,v) = \sum_{w:(v,w)\in E} f(v,w) \qquad \forall v \in V - \{s_0,t_0\} \tag{5.6}$$

$$\sum_{u:(u,v)\in E} f(u,v) \leq c(v) \qquad \forall v \in V - \{s_0,t_0\} \tag{5.7}$$

$$f(u,v) \geq 0 \qquad \forall (u,v) \in E \tag{5.8}$$

5.3 Modelling software

In the previous section we described a method to compute the availability of systems designed by Vanderlande. This method is too complex to compute the availability manually or with the use of spreadsheets. Therefore, we design a software tool, written in Java, to compute the availability automatically.

Figure 5.4 shows the interface of our software tool. On the left side we can find all the input data and parameters, whereas the output is showed on the right side of the screen. Table 5.2 shows the format and example of the input data file. Next to the input data file, the user can define several parameters. The time horizon can be set and the significance and width of the confidence interval can be chosen.

The availability calculation algorithm described in Section 5.2 is programmed in Java. To calculate the flow through the system, as Section 5.2.1 describes, we use optimization software CPLEX Optimizer from IBM. This software is designed to calculate the optimal solution of linear programs. CPLEX itself calculates the optimal solution relatively fast. However, the communication between our software tool and CPLEX is slow. Section 5.4 deals with the running time of our software tool in more detail.

5.4 Running time

We test our software tool using different input structures and parameters. We measure the running time for a serial structure, parallel structure and k out of n-structure. For each structure type, we test systems between 5 and 25 components. Table 5.3 shows the running time for several settings.

ID	Capacity (units per hour)	Predecessors	Successors	MTTF (hours)	MTTR (hours)	Failure distribution	Repair time distribution
1	120		2,3	90	10	exponential	exponential
2	120	1	4,5,6	80	20	exponential	exponential
3	120	1	4,5,6	80	20	exponential	exponential
4	60	2,3		70	30	exponential	exponential
5	60	2,3		70	30	exponential	exponential
6	60	2,3		70	30	exponential	exponential

Table 5.2: Input data file

Number of components	5	10	15	20	25
Serial structure					
Running time (s)	1.26	1.26	5.15	6.10	6.43
Cplex - optimize %	0.58%	1.12%	1.15%	1.23%	1.31%
Cplex - overhead %	52.54%	72.28%	82.03%	89.70%	92.78%
Flow computations	17	60	349	590	763
Parallel structure					
Running time (s)	8.91	14.91	35.40	99.12	171.56
Cplex - optimize %	0.29%	2.18%	3.50%	3.47%	3.62%
Cplex - overhead %	16.27%	55.69%	88.26%	93.16%	94.39%
Flow computations	32	975	7,100	21,878	35,905
k out of n-Structure					
Running time (s)	1.34	15.47	16.70	1048.32	1731.66
Cplex - optimize %	1.20%	2.15%	3.36%	10.44%	?%
Cplex - overhead %	25.06%	56.10%	89.75%	84.49%	?%
Flow computations	32	1,024	28,371	203,750	328,324

Table 5.3: Running time in seconds for serial and parallel system

Regarding the serial structures, the running time increases for an increasing number of components. This is due to the increasing number of flow computations. A major part of the running time is taken by the integrated CPLEX optimizer. The CPLEX optimize algorithm itself is fast, but the overhead time, the communication between the rest of the software tool and the optimizer, is relatively large. The CPLEX overhead takes a major part of the running time. For example, for a serial structure consisting of 25 components, CPLEX optimizer and its overhead takes 94.09% of the running time. With respect to parallel and k out of n-structures, the increase in running time is larger. This is due to the exploding number of flow computations. The overhead time could be avoided by using a programming language where the ability to solve linear programs is included.

5.4.1 Time to specify all inputs

Obtaining the input data takes a relatively long time and has to be done manually. For each component, we need its predecessors and its successors, its capacity and so on. A small time to specify the input data is not within our scope. In the future, obtaining input data could be automatized using the layout of the system.

Component	Failure probability	Repair probability
1	0.0019	0.0165
2, 3	0.0021	0.0083
4, 5, 6	0.0024	0.0055

Table 5.4: Failure and repair probabilities of example system

5.5 Example system

In this section we first apply the simulation algorithm described in Section 5.2. Next, we discuss the results of the simulation algorithm using our modelling software.

5.5.1 Simulation

To show how our method works, we describe the steps in the algorithm using our previous used example system. Figure 5.1 shows this system and Table 5.2 shows the input data for this system. Suppose that we want to compute the expected system availability over a year, with intervals of ten minutes and that the required peak flow of the system is 120 per hour. Furthermore, we set the significance level to 0.05 and the interval width to 0.5. We stepwise show how to do the simulation.

- Start with the initialization: Set the width of the confidence interval to infinity. Compute the failure and repair probabilities for each component. The failure probability p_{f_1} and repair probability p_{r_1} are given by:

$$p_{f_1} = 1 - e^{\frac{-\Delta t}{MTTF_1}} = 1 - e^{\frac{-\frac{1}{6}}{90}} = 1 - e^{\frac{-1}{540}} = 0.0019 \tag{5.9}$$

$$p_{r_1} = 1 - e^{\frac{-\Delta t}{MTTR_1}} = 1 - e^{\frac{-\frac{1}{6}}{10}} = 1 - e^{\frac{-1}{60}} = 0.0165. \tag{5.10}$$

  Likewise we compute the failure and repair probabilities for the other components. Table 5.4 shows these probabilities.

  Next, we compute the maximum flow per time interval. We do this using the linear program which we described in Section 5.2.1. The linear program is described below:

$$\begin{aligned}
&\text{maximize} && f(s_0,1) + f(s_0,2) + f(s_0,3) && && (5.11)\\
&\text{subject to} && f(s_0,1) = && f(1,2) + f(1,3)\\
&&& f(s_0,2) + f(1,2) = && f(2,4) + f(2,5) + f(2,6)\\
&&& f(s_0,3) + f(1,3) = && f(3,4) + f(3,5) + f(3,6)\\
&&& f(2,4) + f(3,4) = && f(4,t_0)\\
&&& f(2,5) + f(3,5) = && f(5,t_0)\\
&&& f(2,6) + f(3,6) = && f(6,t_0)\\
&&& f(s_0,1) \leq && 20\\
&&& f(s_0,2) + f(1,2) \leq && 20\\
&&& f(s_0,3) + f(1,3) \leq && 20\\
&&& f(2,4) + f(3,4) \leq && 10\\
&&& f(2,5) + f(3,5) \leq && 10\\
&&& f(2,6) + f(3,6) \leq && 10\\
&&& f(u,v) \geq 0 && \forall (u,v) \in E.
\end{aligned}$$

The maximum flow per hour is 120. This can easily been seen from Figure 5.1 and the capacities of the components showed by Table 5.2 as well. Component 1 is the bottleneck, with a capacity of 120. Hence, the maximum flow per time interval of ten minutes is 20.

- Add replication 1.
  - Set the time interval $t = 0$. We assume that all components are working at time zero.
    - Start with component 1. Because all components are working at the beginning of the time interval, we compute random numbers for each component and compare them with its corresponding failure probability. We draw a random number for component 1, suppose we draw $RN_1 = 0.6640$. Because this number is larger than the failure probability p_{f_1}, we consider the equipment as working. We draw random numbers for the other components as well and suppose all random numbers are larger than the failure probabilities of its corresponding component. So, at the end of the first time interval, after ten minutes all components are still working.
    - Because we use this first time interval as warm-up period, we are not interested in the flow through the system.
  - Increase the time interval to $t = 1$. All components are working at the beginning of this interval. Again, we draw random numbers for each component and compare them to their failure probabilities. Like the previous interval, all components are working during this interval. Next we compute the flow through the system. Because all components are working, the flow through the system equals the maximum flow of 20. This is equal to the required system flow. Therefore, we consider the system as available.
  - During interval $t = 2$ to $t = 33$ all components are working. The system flow is 20. Hence, the system is available.
  - In interval 34, we draw for the first time a random number that is smaller than the failure probability of its corresponding component, $RN_2 = 0.0007$. This means that component 2 failed during this interval. However, since component 2 and 3 are in parallel, component 3 can handle all the required flow on its own. Hence, the system flow is still 20.
  - In interval 35, we draw a random number for component 2, $RN_2 = 0.5723$. This number is larger than the repair probability of this component. Therefore, the component has not been repaired during this interval.
  - During interval $t = 36$ to $t = 166$ component 2 is still not been repaired. The other components work. The system flow is 20. During interval $t = 167$ component 2 has been repaired and all components are working again. In interval $t = 168$ to $t = 206$ all components are working as well. In interval $t = 207$, component 2 again failed. During interval $t = 207$ to $t = 284$ component 2 has not been repaired.
  - During interval $t = 285$ component 2 has still not been repaired. Moreover, component 3 failed during this interval. As can be seen from Figure 5.1 the flow through the system is zero. The system is unavailable.
  - We continue to simulate the state of the components for the remaining time intervals.
- We count the number of times that the system was available during a time interval. Of all the $365 \cdot 24 \cdot 6 = 52,560$ intervals, 37,732 times the system is available. This gives a yearly availability of $\frac{37,732}{52,560} \cdot 100\% = 71.78\%$.
- Because this is the first interval, we cannot compute a confidence interval. Therefore, the width of the confidence interval remains infinity.
- The width of the confidence interval is larger than the required width of 0.5, so we add another replication. We repeat the same procedure as for the first replication. However, this replication

we have 34,464 times an available system. This gives a yearly system availability of 65.57%. We compute the confidence interval of this two replication. (see Law (2007), Section 9.4 on how to compute a confidence interval). The width of the interval is 79.00. This is larger than the required width of 0.5. Therefore, we add another replication.

- We continue to add replications. The width of the confidence interval gradually becomes smaller. After 409 replications, the width of the confidence interval is, with 0.4993, just below the required 0.5, so we stop. The system availability A_s is the average of the system availability of all replications, $A_s = 67.86\%$. The 95 %-confidence interval is (67.62%;68.12%).

5.6 Requirement fit and verification

In this section we discuss to what extend our model meets the requirements listed in Chapter 4. Next to that we discuss how we verify our availability computation model. A validated model is a theoretical model that meets the requirements. Verification means that we ensure that the theoretical model is correctly translated into the software tool (Law, 2007). First we discuss the requirement fit of the model followed by the verification of our model.

5.6.1 Requirement fit

In Section 4.1 we described five high priority requirements and five low priority requirements for our availability model. To validate our model, we discuss the extend to which these requirements are met by the theoretical availability simulation that we described in Section 5.2. Table 5.5 shows these requirements once more. To start with the first high priority requirement, our availability calculation is suitable for each business unit. The availability simulation is the same for baggage handling systems, parcel & postal systems and warehouse automation systems of Vanderlande. Next, our prototype simulation software computes the expected availability of the whole system. However, a subsystem functionality can easily be added. The simulation algorithm is suitable to compute availabilities of subsystems as well. Third, since the simulation is completely automated, given a certain input and user parameters, the simulation output will be exactly the same regardless of their user. Nonetheless, the user has to make the input data himself. This can result in differences between users. In the future, Vanderlande can automate the making of the input data to avoid this issue. Fourth, the model is easy to explain. It simply simulates the state of each equipment over the complete time horizon and checks at each time if the system can handle the required flow. To support this, Vanderlande could make a simple animation to visually show how the simulation is done. Finally, our method is mathematically correct. We use simulation and availability calculation techniques based on scientific literature. To conclude, our model currently meets four of the five high priority requirements. The fifth can be added later on without much effort.

With respect to the low priority requirements, only two of the five low priority requirements are met. Our current software tool is not very flexible. Due to the long running time, the model does not quickly calculate the change in availability for small layout changes. However, the running time of our algorithm can be reduced once our software is reviewed by software engineers. A small running time is out of scope of this research. Once the running time is reduced, our model will be flexible. The next low priority requirement is a model that can be automated in the future. If the input data can be retrieved automatically, our model can be automated, so this requirement is met. Our availability model currently does not allow for multiple input types, such as check-in and transfer baggage. However, our software is object oriented. This way, a multiple input type feature can easily be added later on. Our model is not completely in line with the availability calculation of installed systems. Nevertheless, the availability calculation of installed system can relatively easy be adjusted to fit our availability estimation. Our model

High priority requirements	Low priority requirements
1. System-standardization: same method for each business unit 2. Output in terms of system availability as well as subsystem availability 3. User-standardization: same answer regardless of the user 4. Easy to explain the model (and its results) to the customer 5. Mathematically correct method	1. Flexible model: allows for quickly calculating the change in availability for small layout changes 2. Model that can be automated in the future 3. Model that allows for multiple input types (for example, check-in and transfer baggage) 4. Model that is in line with availability calculation of installed systems 5. In line with industry standards

Table 5.5: System availability for different system structures

is in line with the underlying theory of the industry standards FEM and VDI. We use the same principles for calculation the availability of a serial and parallel structure.

5.6.2 Verification

Law (2007) describes several techniques to verify a simulation program. We discuss the techniques that we use to verify that our simulation tool performs as expected. Debugging of sub-modules is the first technique mentioned by Law (2007). As described in Section 5.3, we use objective oriented programming for our software. With this way of programming we can easily make use of structured independent building blocks. We debugged each building block individually as well as in relation to the others. Furthermore, we regularly compared the results of a simple simulation for each simulation step with hand calculations. This way we know that the simulation behaves as expected. Finally, we performed many simulations under a variety of settings and checked if the results are realistic. In all situations, the outputs are realistic figures.

5.7 Summary

In this chapter we discussed the following points:

- We need **input data** to do our availability calculation. The input data consist of a data file with information about the structure of the material handling system such as capacities and the failure distribution of the equipment. Furthermore, there are additional parameters, such as the significance of the confidence interval and the time horizon.
- The availability calculation is done by a **simple time simulation** using the probabilities to equipment failure and repair. For each time period, the status of all equipment is simulated. Next, the algorithm computes the flow through the system at that time period. If the flow is higher than the required peak flow, the system is available during that time period.
- The availability calculation algorithm is programmed in **Java** with the use of **linear program optimization software CPLEX**. The running time of our software tool is small for small systems, but rapidly increases for systems with more equipments.
- We **validate and verificate** our availability algorithm. Four of the five high priority requirements are met and two of five low priority requirements are met. It is possible to meet the other requirements in the future. We decribed several ways in which we verificate our method.

Chapter 6

Model validation

In this chapter we discuss the results of our availability simulation technique. In Section 6.1 we discuss results per system structure. Next, in Section 6.2 we discuss the results of the example system which we introduced in Chapter 2. We end this chapter with a discussion of the running time of the software tool and the time it takes to make the input data for the software tool.

6.1 System structures

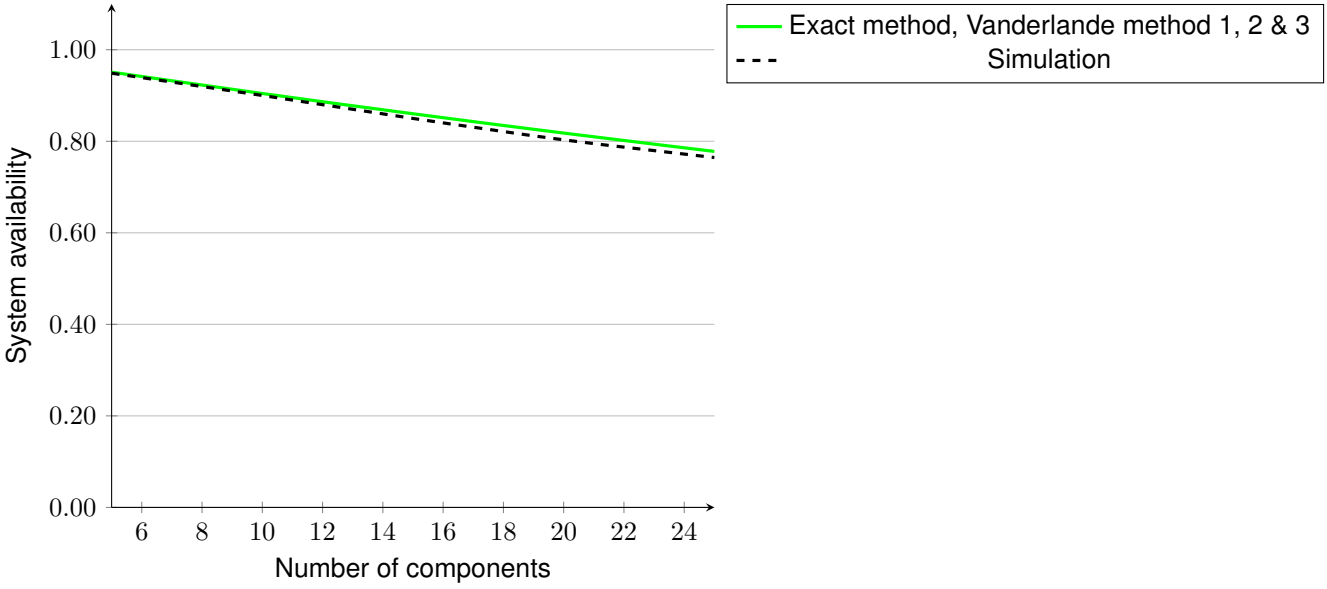
In Chapter 2 we described the serial, parallel and k out of n system structure. In this section we discuss how the availability simulation performs for each of the system structures.

6.1.1 Serial system

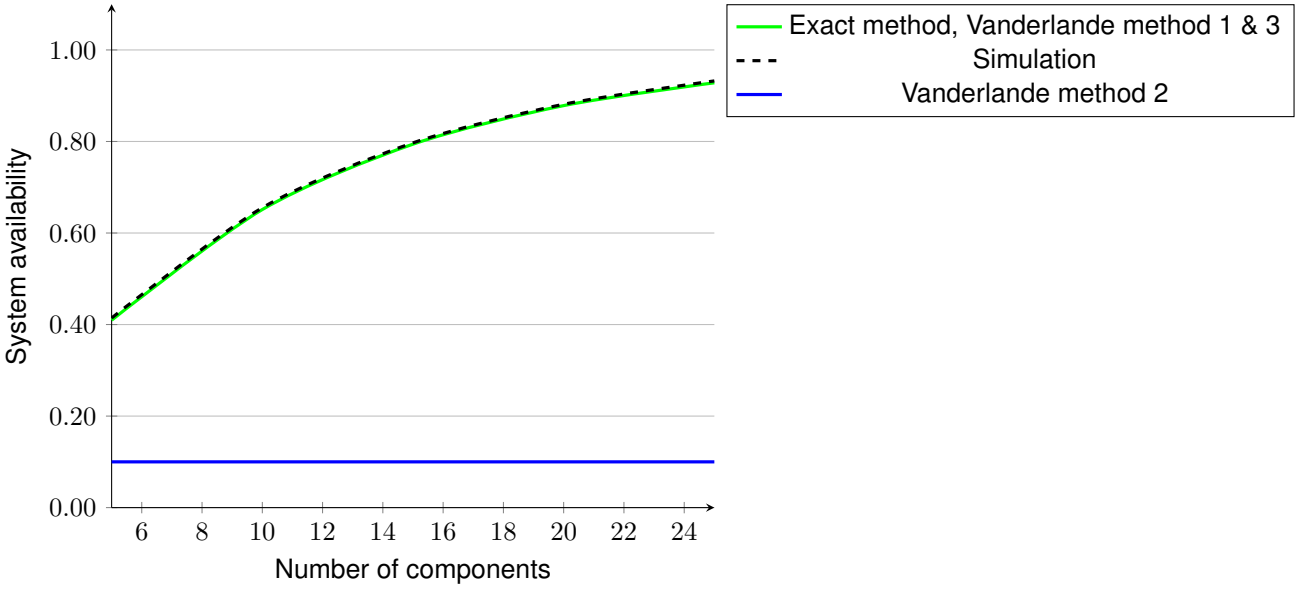
As described in Chapter 2, a serial structure is a system for which all the components need to work. If one of the components fail, the system will be unavailable. Suppose we have a serial system with identical components. The availability of each component is 0.99. For this system we can calculate the availability in an exact way. We described how to do this in Chapter 2. Suppose that we are interested in the long term availability. We should then take long time horizon; suppose we choose a year. Furthermore, set the significance level to 0.05 and the width of the confidence interval to 0.5. Table 6.1 and Figure 6.1 show the availabilities for this system for varying numbers of components for the exact method and for our simulation. We are aware of the fact that the systems of Vanderlande consist usually of much more than 25 components. However, our software tool currently cannot compute the expected availability for such large systems within reasonable time. We discuss the running time of the tool in Section 5.4. From Table 6.1 and Figure 6.1 we see that the simulation is accurate for serial systems with a small number of components. For larger systems the simulation tends to underestimate the availability. This is in contrast to the currently used methods by Vanderlande. Those serial structure availability computations are equal to the exact method.

6.1.2 Parallel system

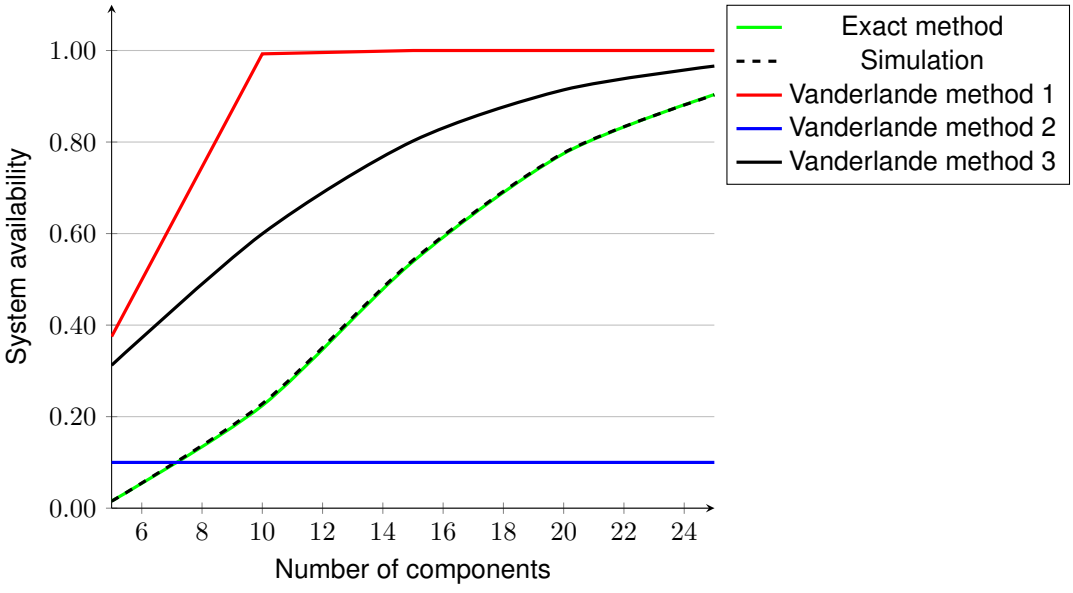
Recall from Chapter 2 that for a parallel system structure only one of the components need to work to have a working system. Again, consider a system of identical components. But now the component's availability is 0.1. We keep all other parameters the same. Thus, a significance level of 0.05, confidence interval width of 0.5 and a time horizon of one year. Table 6.1 and Figure 6.1 show the availabilities for this system for varying numbers of components for the exact method and our simulation. We see that our simulation is in line with the exact method, but it slightly overestimates the system availability.



(a) Serial structure



(b) Parallel structure



(c) k ouf of n structure

Figure 6.1: System availability of the three basic system structures

Components	Serial structure		Parallel structure		k out of n	
	Exact method	Simulation	Exact method	Simulation	Exact method	Simulation
5	0.9510	0.9487	0.4095	0.4143	0.0156	0.0152
10	0.9044	0.9001	0.6513	0.6552	0.2241	0.2283
15	0.8601	0.8499	0.7941	0.7970	0.5387	0.5419
20	0.8179	0.8031	0.8784	0.8813	0.7748	0.7763
25	0.7778	0.7645	0.9282	0.9323	0.9038	0.9029

Table 6.1: System availability for different system structures

A_i	Method 1	Method 2	Method 3	Exact method	Simulation
A_S	0.7940	0.5040	0.7590	0.6774	0.6786
A_{IE1}	0.9600	0.8000	0.9600	0.9600	0.9587
A_{IE2}	0.9190	0.7000	0.8785	0.7840	0.7853

Table 6.2: Availability of example system

Method 1 and 3 of Vanderlande are equal to the exact method and thus performs slightly more accurate. However, in Chapter 3 we already saw that method 2 shows a huge deviation from the exact method. The system availability is 0.1 for all number of components according to this method.

6.1.3 k out of n-System

From Chapter 2 we know that a k out of n structure is similar to a parallel structure. The system is available if k of the n components work. We consider a system of identical components, with component availability 0.25. Suppose at least four components need to work to have a working system. Thus, $k = 4$. All other parameters remain the same as for the previous system structures. Table 6.1 and Figure 6.1 show the availabilities for this system against the number of components. Our simulation is in line with the exact method. In contrast, the methods that Vanderlande currently uses show huge deviation from the exact method. Method 1 and 3 clearly overestimate the system availability, while method 2 shows a constant availability of 0.1.

6.2 Example system

In this section we discuss the results of the simulation of the example system. Figure 5.1 shows this system. This system is a combination of the three basic system structures. Table 6.2 shows the availability of the example system calculated with the methods of Vanderlande, the exact method and our simulation technique. The table not only shows the system availability (A_S), but the availability for the two subsystems IE1 and IE2 as well. Recall that subsystem IE1 is a parallel structure and subsystem IE2 is a k out of n structure. Section 2.2.1 describes these subsystems in more detail. As can be seen, our results are close to the availabilities of the exact method. For the whole system and subsystem IE2, our simulation technique clearly outperform the current methods used by Vanderlande. Regarding subsystem IE1, our simulation technique result is close to the availability of the exact method. However, the availability of method 1 and 3 are equal to the exact method availability. Hence, our simulation technique performs less well for subsystem IE1 compared to the first and third method currently used by Vanderlande. This is caused by the fact that the simulation approximates the availability, whereas Vanderlande's method 1 and 3 use exact calculations for parallel structures such as subsystem IE1. Figure 6.2 shows the availability of the example system with six identical components for varying component availabilities. In contrast to the methods currently used by Vanderlande, we see that our simulation is in line with the exact method.

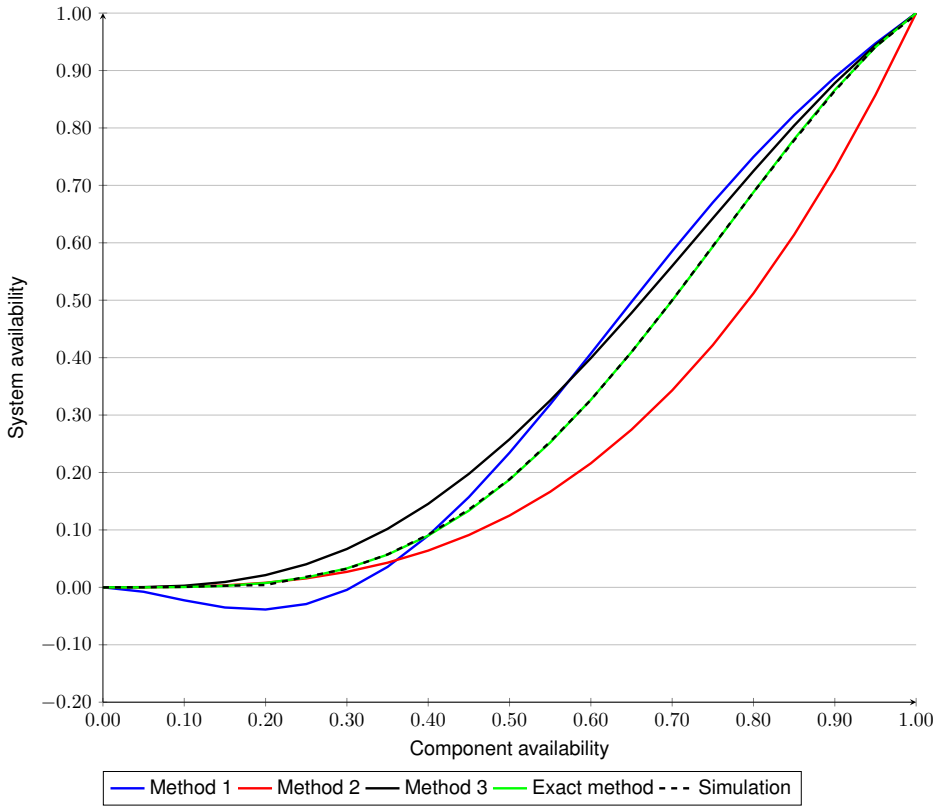


Figure 6.2: System availability of example system

A_i	Day	Week	Month	Year
A_S	0.8779	0.7200	0.6883	0.6786
Replications	27,123	16,617	4,737	409

Table 6.3: Availability of example system with varying time horizon

When the time horizon becomes smaller, we see that the expected availability increases. Table 6.3 shows the availability of the example system during a day, week, month and a year. Furthermore, the table shows the required number of replications as well to get a 95%-confidence interval with a with of 0.005. We see that we need more replications for smaller time horizon. This is because the results of one replication are more unstable compared to replications with a longer time horizon. The increasing availability for smaller time horizons is due to the smaller probability of a component failure during one replication.

Next to varying the time horizon, we vary the significance level as well. We used again the example system with a time horizon of a year. We set the width of the confidence interval to 0.05. The system availability tends to decrease for decreasing significance level. As expected, the results are closer to the exact method when the significance level decreases. We need more replications to obtain the more accurate result.

Finally, we perform the simulation for varying confidence interval widths. As for the significance level, the system availability decreases as the width of the interval becomes smaller. However, for a interval width of 0.001 the system availability is estimated smaller than calculated with the exact method.

Significance level	0.01	0.05	0.10
A_S	0.6783	0.6786	0.6794
Replications	690	409	298

Table 6.4: Availability of example system with varying significance level

Confidence interval width	0.001	0.005	0.01
A_S	0.6769	0.6786	0.6820
Replications	10,078	409	94

Table 6.5: Availability of example system with varying interval width

Chapter 7

Conclusion, discussion & recommendations

With this chapter we end this thesis. We start with conclusions about our research in Section 7.1. Next, in Section refsec:discussion we discuss what our findings might mean. How valuable are they and why? We end this chapter with Section 7.3 where we give recommendations to further develop and implement our availability tool for Vanderlande.

7.1 Conclusion

In this research we designed a model to calculate the expected availability of a material handling system to be build by Vanderlande. A system is unavailable when due to technical failure the system can not meet the throughput where Vanderlande and its customer have agreed on' (Vlasblom, 2009). The purpose for this availability model is to use it during the design phase to see if the designed model meets the customer's requirements and to use the expected availability in service contracts. We expect more accurate availability estimations and time savings in calculating this.

We examined five methods to estimate system availability. In agreement with Vanderlande, we defined a priority ranked list of requirements. Based on these models and requirements we proposed three alternative ways of availability calculation. We decided to use a simple time simulation to model the system availability. This because it is a relatevely simple technique giving accurate results.

Our model loops over the user defined time horizon and simulates in each time period the state of each equipment. By knowing the state of each equipment, the system state is obtained. We count the number of times the system is working and calculate the availability by dividing this number by the number of time periods. We repeat this process until we obtain the desired level of accuracy.

We found that our model is accurate for parallel and k-out-of-n structures. For serial structures our model underestimates system availability compared to the exact serial system availability. A combination of different structures gives accurate results.

7.2 Discussion

The first matter of discussion is the deviation of the serial structure. For a serial structure consisting of 25 components with identical availability of 0.99, the system availability deviates 1.71 percent from the system availability obtained by an exact calculation. We are not sure how much the system availability

deviates for larger serial structures. Furthermore, we do not know why this deviation exists. It is possible that the simulation is not as accurate as we expect.

The second discussion point is that we do not know how the simulation performs for large and complex systems, such as the material handling systems of Vanderlande. The software is not powerful enough to simulate such systems.

Third point of discussion is the warm-up period. In our model, we only use the first period as warm-up period. One period might not be enough to obtain a steady-state system.

7.3 Recommendations

We have several recommendation to continue this research and develop our availability model to an operational software tool.

First we recommend to make sure that the input data is reliable, up to date and automatically retrieved. By reliable and up to date input data more accurate and reliable results will be obtained. We recommend to implement a data feedback loop. Comparison between availability estimation and realized availability data of installed systems and components can be used to improve the availability simulation. One can think of using probability distributions that better represent the failure and repair behaviour of the equipment. Once the input data is automatically retrieved, the software tool is more likely to be used in practice. Furthermore, time is save which can be used for other more useful tasks. Once reliable and up to date data is achieved,

Second, our model can only process small systems within reasonable time. We recommend to further develop our software tool to make it suitable for large and complex systems as well. This can be done, among others, by improving the integration of the flow computation algorithm software into the software tool. A mathematician or information scientist, whether or not a graduate, could come up with more efficient techniques to reduce the running time of the model. Furthermore, an user friendly layout has to be designed. Some relevant features could be added, such as componant importance, availability calculation of subsystems.

Third, we recommend to investigate whether or not one warm-up period is appropriate. This can be done using Welch's procedure decribed in Section 9.5 of Law (2007). The results of our model would be more reliable if an appropriate warm-up period is used.

References

European Federation of Materials Handling. (1989). *Standards of the acceptance and availability of installations with storage/retrieval machines and other machinery* (FEM No. FEM 9.222).

Heerkens, H., & Van Winden, A. (2012). *Geen probleem*. Business School Nederland.

Iyer, S. M., Nakayama, M. K., & Gerbessiotis, A. V. (2009). A markovian dependability model with cascading failures. *IEEE Transactions on computers*.

Korver, B. (1994). *The monte carlo method and software reliability theory.*

Kuo, W., & Prasad, V. R. (2000). An annotated overview of system-reliability optimization. *IEEE Transactions on Reliability*.

Kuo, W., & Wan, R. (2007). Recent advances in optimal reliability allocation. *Computational Intelligence in Reliability Engineering*.

Lassche, M. (2015). *Availability calculation standard for material handling systems.*

Law, A. M. (2007). *Simulation modeling & analysis*. Mc Graw-Hill.

Lu, H., Jin, X., & Chang, H. (2012). A monte carlo method of complicated system availability analysis using component state trace. *IEEE Quality, Reliability, Risk, Maintenance, and Safety Engineering*.

Manthey, B. (2015). *Optimization modeling [lecture notes]*.

Mettas, A. (2000). Reliability allocation and optimization for complex systems. *Reliability and Maintainability Symposium*.

Modern Material Handling. (2015). *Top 20 systems suppliers, 2015.* Retrieved 2015-06-03, from `http://www.mmh.com/article/top_20_systems_suppliers_2015`

Rausand, M., & Hoyland, A. (2004). *System reliability theory*. Wiley Interscience.

Ross, S. M. (2007). *Introduction to probability models*. Academic press.

Takeshi, M. (2013). A monte carlo simulation method for system reliability analysis. *Nuclear Safety and Simulation*.

Topuz, E. (2009). Reliability and availability basics. *IEEE Antennas and Propagation Magazine*.

Vanderlande Industries. (2014). *Annual report 2014.*

Vanderlande Industries. (2015a). *Baggage handling.* Retrieved 2015-06-03, from `https://www.vanderlande.com/baggage-handling/`

Vanderlande Industries. (2015b). *Life-cycle services.* Retrieved 2015-06-03, from `https://www.vanderlande.com/life-cycle-services/`

Vanderlande Industries. (2015c). *Parcel and postal.* Retrieved 2015-06-03, from `https://www.vanderlande.com/parcel-and-postal/`

Vanderlande Industries. (2015d). *Warehouse automation.* Retrieved 2015-06-03, from https://www.vanderlande.com/warehouse-automation/

Verein Deutscher Ingenieure. (n.d.). *Anwendund der verf˙*

Vlasblom, R. P. (2009). *Steering life cycle costs in the early design phase.*

Winston, W. L. (2003). *Operations research*. Cengage Learning.

Zhang, X. (2014). A multi-dimensional component importance analysis mechanism for component based systems. *IEEE Annual International Computers, Software and Applications Conference*.