Modelling Degradation of Physical Objects in Fault Maintenance Trees

Tycho Braams
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
t.l.braams@student.utwente.nl

ABSTRACT

In an ideal world, systems never fail. Since we do not live in an ideal world it is necessary to maintain systems to repair failures. A maintenance strategy is used to determine how often a system is inspected and when components need to be repaired or replaced.
A Fault Maintenance Tree (FMT) is a model used to compare maintenance strategies. FMTs are used to find an optimal maintenance strategy. A maintenance strategy is meant to decrease the number of failures that cause downtime. Maintenance itself also incurs costs and planned downtime for repairs. An optimal maintenance strategy needs to find a balance between both costs.
To be able to give reliable predictions for maintenance strategies, the model has to be as realistic as possible. If the model predicts a certain failure rate, this needs to match the failure rate seen in the actual system.
This paper studies the way degradation of electrically insulated railway joints is modelled. This is done using the model developed in a previous study. This resulted in the discovery of several anomalies in the model, as well as the previously hidden effect of repairs on the system.

Keywords

Fault Maintenance Tree, Degradation, Maintenance strategy

1. INTRODUCTION

Two important factors for the performance of any physical system are the reliability of the system and the number of failures of the system. To improve the reliability of a system and decrease the number of failures, maintenance strategies are created and implemented. For a maintenance strategy to be successful it is important to first know how and when a system will fail, and then to know how maintenance will affect such failures.
Fault Tree Analysis (FTA)[7] is widely used to analyse physical systems. If the failure rates of the components of the system are known, the odds that the entire system will fail can be calculated. However, it is not possible to include the effects of maintenance on such a model. As such, FTA is not a suitable tool to find the effects of different maintenance strategies.

28^{th} *Twente Student Conference on IT* February 2^{nd}, 2018, Enschede, The Netherlands.

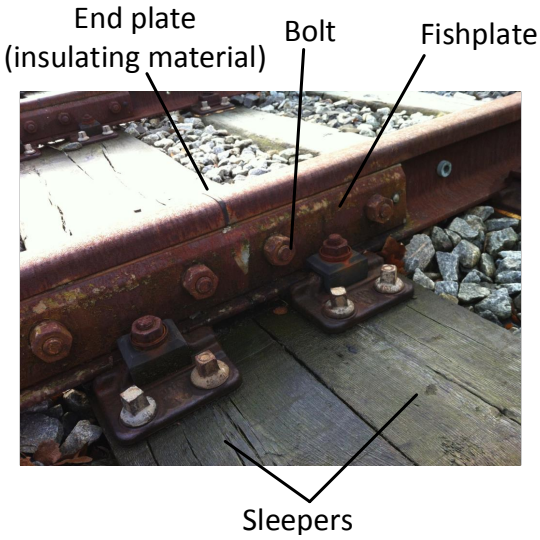

Figure 1. An electrically insulated joint with the visible components indicated.[6]

Fault Maintenance Trees (FMTs)[5] are being developed to simulate different maintenance strategies. An FMT extends fault trees to include component degradation, inspections and repairs.
Previous work by Ruijters et al.[6] focused on modelling Electrically Insulated joints (EI-joints, see Figure 1) using FMTs. EI-joints separate railway tracks into segments and detect if a train is located on the segment. EI-joints contain components that fail without warning and components that gradually decrease in functionality.
The model developed by Ruijters et al. was reported to be created with data, originally obtained from ProRail, as well as data obtained by interviewing experts. The model created realistic predictions for the failure of the electrical components but the predictions for the physical components showed some deviation from reality.
This paper will focus on analysing this model using the Uppaal tool[3]. First, the results of the previous study will be reproduced. Then, we identify several avenues for improvements to the model.

1.1 Research Question

The research will focus on the main research question.

Research Question: Can the model for EI-joints from [6], using Fault Maintenance Trees, be improved?

To answer this question, the following subquestions have been formulated.

Question 1: Can the results obtained in the study by Ruijters et al.[6] be reproduced?

Question 2: What is the effect of maintenance on the predicted failures?

1.2 Research method

To answer the subquestions and subsequently the main research question, the following steps were followed:
First, a literature study was performed to study FTA and FMTs and other possible modelling solutions. Furthermore, the behaviour of metal degradation was researched. The Uppaal tool was also studied.
After the literature study, the results presented in [6] were reproduced. Then, attempts were made to model continuous degradation. To complete this step, a rate of degradation was needed. This was calculated using the MTBFs stated in [6]. Based on these calculations the number of repairs performed by the model was studied. Furthermore, the number of performed inspections was studied to find the reason for the repairs. This required running several simulations with the current model.

1.3 Motivation

Although the failures predicted in the previous study mostly matched the failures observed in actual joints, a few failure types showed significant deviations. A hypothesis to explain the deviation was that the way the degradation was modelled in the FMT and Uppaal, using a set of discrete phases, did not do justice to the degradation taking place in the real world. The proposed solution to this problem was to change the model for the objects showing large deviations to incorporate continuous degradation instead of using discrete phases.
The discrete phases of an object could be seen as different states of an object. An example of an object with four phases would be: New, used but no visible faults, visible faults but no failure, and failed. While these are all reasonable qualifiers for the phase of an object, if these are used as discrete phases, this implies a clear, concrete moment where an object moves from one phase to another. If an object is associated with a continuous degradation, such an object would not have such clear phase changes.

The approach to incorporate continuous degradation was to move from the phases to a continuous value, indicating the degree of degradation for an object. This can be compared to the system of using a number of hitpoints to indicate the health of a player in many video games. If the variable recording the degradation was to be implemented using an integer, this would not create the desired effect. Since an integer variable can only contain a discrete set of values, this would effectively just be increasing the number of phases. This is not necessarily an impossible approach. When looking at mathematics, using a limit to decrease the step size and increasing the number of steps can be used to approximate a continuous function. This is not the chosen approach since increasing the number of steps taken will improve the complexity of the model and very likely increase the time taken to run simulations.
Since the clocks in Uppaal are continuous values, these lend themselves very well to recording a continuous variable. To correctly model degradation using a clock, the clock needs to be set to an appropriate rate. This rate should be based on the time it takes for a component to fail. Calculations to find such a rate led to the discovery of the results presented in this paper.

2. BACKGROUND

In this section, background information will be provided to give a context to the concepts used in the paper. We first introduce degradation of components in Section 2.1. In Section 2.2 we introduce the terms used for expected failure values. We then explain fault tree analysis in Section 2.3.
Lastly we explain Uppaal and the Monte Carlo technique used by Uppaal in Section 2.4 and Section 2.5.

2.1 Degradation

Physical degradation can be split into two categories: objects with a sudden point of failure and objects with a gradual loss of function.
An example of an object with a sudden point of failure is a light bulb. A light bulb produces light when turned on up to a moment when it stops. Some light bulbs start flickering before losing total functionality, but there is no maintenance strategy to influence the lifetime of a light bulb. Sometimes light bulbs have production errors. Therefore it was customary to test your light bulb in the shop before payment. This pattern, with early failure and end-of-life failure, is often referred to as a bathtub pattern[8][9].
An example of an object with a gradual loss of function is the brake system on a bicycle. The brake pads are slowly worn down until the brakes no longer influence the speed of the bike significantly. The settings for the brakes can be changed so that the brake pads are closer to the wheel. The brakes will thus regain some of their functionality. Eventually, the brakes will be worn down too much and will have to be replaced. There is often an indicator on the brake pads to show when this point has been reached. This indicator is based on the average lifetime for brake pads.
It is necessary to account for the consequences of a failure when deciding on a maintenance strategy. Where a broken light bulb leaves you in the dark until the bulb is replaced, if the brakes on a bicycle do not work you could end up on the bonnet of a car. The failure of an object can also have different consequences based on the location of the object. A light bulb responsible for illuminating a staircase should be replaced as soon as possible, while a light bulb in a room with multiple sources of light can be replaced with less priority.
For an object with a sudden point of failure, the moment to repair the object is clear. For an object with a gradual loss of function, there might be an indication when it should be replaced. To decide the repair moment based on such an indicator it is important to know the standard deviation. A large standard deviation could lead to a large under- or over-estimation for the repair moment and as such an increase in the cost of the maintenance strategy. A replacement based on such an indicator can be seen as a preventive replacement meant to prevent total failure.
Based on the consequences of a failure it can be decided if it is necessary to perform a preventive replacement or a replacement after the object has failed. The costs of a failure and the costs of repairing or replacing an object are important factors to consider when deciding whether to use preventive or reactive measures.
For a maintenance strategy, one can assume that replacing an object with a sudden point of failure is a simple operation. Objects with a gradual loss of function often have a simple method to repair the functionality. However, if the object needs to be replaced this often requires a more complicated action.
A simple operation to replace an object would appear to conclude with low downtime. If it takes a long time for a mechanic to reach the location of the object or there is a limited amount of time available to replace the object it could still lead to a long downtime.
Another important factor for modelling degradation is how

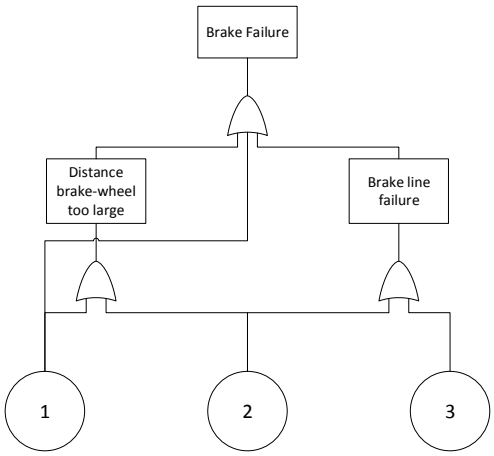

Figure 2. A fault tree for the brakes on a bicycle. 1 indicates worn brake pads, 2 indicates a stretched brake line and 3 indicates a broken brake line

the data about the degradation is obtained. If a log is kept detailing the use of the object, this could be used to improve the failure predictions. However, registering this data will probably introduce errors if this is not done properly. Furthermore, a new cost is created, namely the cost of maintaining the data about the use of an object.
Lastly, a maintenance strategy can be influenced by the number of objects that are maintained. For example, a housing corporation needs to maintain a considerable number of light bulbs in the general areas of their buildings. If these light bulbs are installed at the same time, they will all have the same average lifetime. If a light bulb fails far before the expected lifetime it will lead to the single light bulb being replaced when the corporation is notified by a tenant. Once the average lifetime has elapsed, it will be cheaper to replace all the light bulbs in one large operation than to wait for every light bulb to fail. Every failure would then lead to a single replacement bringing a substantial amount of overhead costs compared to replacing all the light bulbs at once.
In case the brake line fails on the bicycle, a replacement of the brake line will also include replacing the brake pads and re-adjusting their position. If the brake line is of low quality, the likelihood of the brake pads reaching their end of life decreases.
It is difficult to decide on a maintenance strategy when a new object is introduced without an expected lifetime. At first, it is unknown whether a failure is a single occurrence or the precursor for the end of the lifetime.

2.1.1 Metal degradation

Metal objects that are subjected to cyclic loading suffer from metal fatigue. The repeated stress placed on an object causes progressive and structural damage. To calculate the effect of metal fatigue on degradation rates, very specific data is required. It became clear that this data was not available within the scope of this project.
Ahktar et al.[1] performed a study where such calculations were made.

2.2 MTTF

Failure rates of mechanical systems are often quantified using a Mean Time To Failure (MTTF)[1] for nonrepairable systems or a Mean Time Between Failures (MTBF) for repairable systems. These are associated with the time that is expected to elapse before a failure will occur. The Mean Time To Repair (MTTR) is often used to indicate the mean time between failure and repair. These values are often calculated as an arithmetic mean, to account for the variance that is inherent in such values. These calculations use historical data and knowledge about the materials used in creating the object.

[1]Sometimes Estimated Time To Failure (ETTF) is used.

2.3 FTA/FMT

A Fault Tree models the behaviour of a system by combining the failure rates of the components using Boolean gates. An example of a simple Fault Tree (FT) modelling the brakes on a bicycle can be found in Figure 2. All gates in this figure represent logical disjunctions (OR-gates).
In this FT it can be seen that the brakes can fail because either the pads are worn out, the distance between the pads and the wheel has become too large or the line connecting the pads to the handle has failed.
In this model, all the basic events are either functioning or broken. It is clear that this is not an accurate representation of the phase the brake pads are in. This model also can not show the effects of changing the settings of the brakes. As such, Fault Tree Analysis is a very useful tool to find the most likely cause for failures, but it is less suitable to test different repair strategies.
FMTs are an extension for FTA to allow degradation and maintenance to be modelled. The basic events are redefined for degradation. This is achieved by splitting a basic event into multiple states. Every state corresponds to a pre-defined degradation phase of the component. Furthermore, a maintenance model is introduced and thresholds are set on the degradation states to specify when an action needs to be performed. If a component can be repaired this is modelled by setting a basic event back to a previous phase.
FMTs also introduce a new gate to the fault tree: the RDEP (rate dependency). When a component fails, it can sometimes influence the failure rate for related components. This is modelled using the RDEP gate.

2.4 Uppaal

Uppaal is a tool developed by Upsalla University, Sweden and Aalborg University, Denmark since 1995[2]. This tool can be used to analyse, validate and verify real-time systems modelled as networks of timed automata. For the analysis of FMTs, the extended version of Uppaal, Uppaal SMC[3][4] is used. This extends regular Uppaal with statistical model checking.
In order to analyse an FMT with Uppaal, the tree needs to be reinterpreted into a set of priced timed automata (PTA). Every leaf of the tree is changed into a PTA that models the failure behaviour of that leaf. These individual PTAs are then combined to form a network that models the behaviour of the tree.
A PTA is an automaton made up of states and transitions. It is possible to include clocks, which makes the automaton timed. Such clocks can be used in invariants to limit the time that can be spent in a certain state. It is also possible to use the value of clocks in guards of transitions. The value of a clock will then influence which transition is taken out of a state.
When an invariant with a clock is used to limit the time spent in a state, the actual time-delay that occurs in a simulation follows a uniform distribution. If such an invariant is not present, the state has an unbounded delay. The state can then be given an exponential rate. The time-delays will then follow the exponential distribution specified by such an exponential rate.

States can also be made urgent or committed. In both urgent and committed states, no time can elapse. The difference is that while it is possible to take other transitions when there is an urgent state active in one of the PTAs, if a committed state becomes active the only permissible action is to take a transition out of that state.
To model interaction between PTAs, it is possible to use synchronisation channels. To use a synchronisation channel, a transition in one PTA is marked as the sender, while a transition in one or more other PTAs is marked as a receiver. When a sending transition is taken in a PTA, all other PTAs that use the same synchronisation channel must take a transition marked as a receiver.
Uppaal can analyse the set of PTAs with statistical model checking, a Monte Carlo simulation technique. It is possible to define queries using a simplified version of Timed Computational Tree Logic (TCTL). These queries can be used to define reachability properties, safety properties and liveness properties. Furthermore, it is possible to define queries that can answer questions about probabilities.
Reachability properties are used to determine if it is possible to satisfy a given property. This does not guarantee that the property is satisfied in every simulation. A positive result means that there is a path possible that satisfies the property while a negative result indicates that the property can never be satisfied. This can be used to verify that the model can show the behaviour that you are trying to simulate.
Safety properties are used to determine if a certain property will never be satisfied. It is also possible to use safety properties to test if a property will possibly never be satisfied. To test if a property will never be satisfied, the universal quantifier is used. If the existential quantifier is used, it tests if there exists a maximal path where the property will never be satisfied.
Liveness properties are used to test if something eventually happens. It can be used to test if a property is eventually satisfied, but more often it is used to test the link between two properties. Such a link is often a *leads to* or *response* property, where, if A becomes true, then eventually B will become true.
These three properties can be used to verify the correctness of a model or to check if possibly unwanted behaviour is possible in the model, or perhaps the system it is based upon.
Of particular interest for FTA and FMT analysis are the *probability queries*. These can be used to find the probability of a certain property, to compare the probability of a property to a threshold or to compare the probability of one property to the probability of another. Uppaal SMC also supports the evaluation of *expected values*. Given an expression that evaluates to a clock or an integer property, it will calculate the average minimum or maximum value of a set of simulations. For example, it might be relevant to find the average value of a counter in a PTA. The maximum value of the counter will then be the value of the counter at the end of the simulation, given that it is not reset. The average of the maximum values of a set of simulations is thus the average value of the counter.
These queries can also include a quantifier for how long a simulation should last. Queries that evaluate expected values also need to indicate how many simulations should be run. For probability queries, Uppaal will run simulations until it has obtained a statistically relevant result based on parameters such as a Confidence-Interval or the probability of false negatives.

2.5 Monte Carlo

Monte Carlo simulations can be used when a model contains unknown variables. A Monte Carlo simulation involves drawing a large number of pseudo-random uniform values from an interval, such as a $[0, 1]$ interval. These values are assigned meaning based on a stationary probability distribution. For example, if flipping a coin was to be simulated in such a way, any value less than or equal to 0.50 would be heads and values greater than 0.50 would be tails. By the law of large numbers, drawing many random values will lead to the simulation of the behaviour of repeatedly flipping a coin. If a model contains unknown inputs, but a probability distribution for these values, the Monte Carlo technique can be used to approximate these variables.
Because of the pseudo-random values used in Monte Carlo simulations, the results show slight variances. If a larger number of simulations is used to obtain average values, this variance can be reduced. Of course, running more simulations takes more time.

Monte Carlo calculations often involve Markov chains. In a Markov chain, the probability of possible events depends only on the current state. This is sometimes characterized as memorylessness, where information about the history prior to the current state has no influence on the probability of future states or events.

3. RESEARCH

In this section, the research will be explained. First, we present our outcomes when attempting to reproduce the results in Section 3.1. Then, the expected failures are explained in Section 3.2. Then the difference between expected and observed failures is explained in Section 3.3. Finally, the changes made to the model are explained in Section 3.4.

3.1 Reproducing results from the previous study

To be able to verify if changes made to the model from the previous study[6] are improvements or deteriorations, it is necessary to first reproduce the results from the previous study.
The model managed to reproduce the results from the previous study[2].

3.2 Expected failures

The previous study[6] presented a table with MTBFs for every component.
When looking at the MTBF of the first component, poor geometry, this has an estimated failure time of 5 years. The previous study used a simulation time of 50 years. On average, the first component would thus fail 10 times. The previous study also used a population of 50.000 joints. Because only 10% of the joints can experience the first failure type, this would mean 5.000 joints experience this failure. 5.000 joints experiencing 10 failures in 50 years would mean 50.000 failures.
However, the previous study stated that 110 failures were predicted by the model and only 48 failures were observed. When this is calculated for all the failure types, the results in Table 1 are obtained.
For BE11 to BE14, the predicted number of failures is comparable to the calculated number of failures. However, the expected number of failures based on the MTBFs showed a large discrepancy for BE1 to BE10. These values differed from the reported failures with a magnitude of

[2]For all the basic events except for BE8, no explanation has been found yet for this anomaly.

Table 1. Comparison of the number of failures based on MTBF calculation and the number of failures reported in [6].

BE nr.	Failure mode	ETTF (yrs)	Failure Prob.	#Failures in 50 yr	#Failures per year	Predicted	Actual
1	Poor geometry	5	10%	50.000	1.000	110	48
2	Broken fishplate	8	33%	103.125	2.062	129	83
3	Broken bolts	15	33%	55.000	1.100	2.3	2.1
4	Rail head broken out	10	33%	82.500	1.650	68	30
5	Glue connection broken	10	33%	82.500	1.650	70	37
6	Battered head	20	5%	6.250	125	3.4	5.5
7	Arc damage	5	0.2%	1.000	20	7	3.4
8	End post broken out	7	33%	117.857	2.357	12	9.4
9	Joint bypassed: overhang	7	100%	500.000	10.000	212	200
10	Joint shorted: shavings (normal)	1	12%	300.000	6.000	156	150
11	Joint shorted: splinters	200	3%	12.500	250	254	261
12	Joint shorted: foreign object	250	100%	10.000	200	199	200
13	Joint shorted: shavings (grinding)	5000	100%	500	10	10	10
14	Sleeper shifted	5000	100%	500	10	10	18
15	Internal insulation failure	5000	100%	500	10	n.a.	n.a.

$10\times - 100\times$, a much more significant difference than the difference between predicted and actual failures, at most $2\times$.

3.3 Failure prevention

The number of failures calculated with the MTBFs is compared to the number of failures reported by the previous study in Table 1.
The previous study also notes that ProRail reported approximately 3000 joints are replaced every year.
The number of failures obtained from the calculations using the MTBF is the number of failures that will occur if *no* maintenance is performed.
The failures reported in the previous study are the failures that occurred while maintenance to *prevent* failures was performed.
The model includes periodic inspections that can detect degradation and repair this degradation before a failure occurs. The number of failures reported is obtained by counting the number of times an object actually fails, despite inspections taking place. A maintenance action can increase the lifetime of an object when a repair is done, or replace the entire joint, effectively resetting the MTBF for *all* components.
A repair or replacement action is performed in the model by setting the phase of the component back to the first phase. This means that the expected number of failures based on the MTBF is reduced every time a maintenance action is performed. This makes it harder to draw conclusions about the accuracy of the degradation modelling. To be able to determine if a model accurately represents the degradation of an object, it could be useful to obtain accurate data about the failure of joints without maintenance. Furthermore, a different model could be created that models the degradation of the object. This can then be combined to verify that the model accurately models the degradation. When an accurate degradation model is achieved, this can then be incorporated into a model which includes maintenance.
Special cases are the components modelled using a single phase. The expected number of failures closely matches the observed number of failures for these components. In the model, these components cannot trigger a maintenance action before they have failed. The MTBF for these components is still reset by replacing the entire joint, but the MTBF is so much longer than the simulation duration, a reset will have a negligible effect.
Questions could also be asked about the origin of the MTBF values for these components. If these are calculated based on the observed number of failures, it is logical that the expected number of failures matches the observed number of failures.

3.4 Hidden repairs

The previous study only reported the number of failures and analysed the cost of the inspections and repairs. It has become clear that a lot of repairs are executed that prevent failures. As such, it is interesting to find out how many repairs were actually executed, how many replacements were executed and which component triggered these repairs.
To achieve this, several counters were added to the model:

1. In the inspection model, to find how often inspections triggered repairs.
2. In the repair model, to find how often a repair was performed.
3. In the models for the components, to find how often they were repaired before they failed.
4. In the models for the components, to find how often they reached the threshold where an inspection will lead to a repair.

Queries were formed to find average values for each of these counters, for each component in the system, as well as for every inspection module and repair module. These queries were evaluated using 1.000 simulations each lasting 50 years. Some components share an inspection module. This means that if one component indicates that a failure will lead to a repair, the other components associated with that inspection module will also be evaluated as needing repair. Components that share an inspection module often also share a repair module. This means that when one component is repaired, all other components associated with that repair module will also be repaired.
The repair model allows for scheduled repairs. This can trigger a repair every X years, regardless of whether an inspection has triggered such a repair. In the current version of the model, no scheduled repairs are executed, because ProRail indicated they did not use scheduled repairs. This

Table 2. The number of times the different inspection modules triggered a repair per year.

Inspection id	Inspecting BE	#Repairs triggered		Repaired by
I1	BE1	1.263	±231	R1
I2	BE2, BE4, BE15	3.435	±302	R2
I3	BE6, BE7, BE9	2.305	±274	R3
I4	BE10, BE13	1.861	±310	R4
I5	BE5	1.129	±111	R5
I6	None	None		None
I7	None	None		None
I8	BE14	10	±3	R8
I9	BE11	260	±32	R9
I10	BE8	2.624	±246	R11

Table 3. The number of times the different repair modules performed a repair.

Repair id	Repairs BE	#Repairs performed	
R1	BE1	1.363	±236
R2	All except BE12	3.669	±336
R3	BE6, BE9	2.359	±284
R4	BE10, BE13	1.826	±325
R5	BE5	1.247	±121
R6	None	None	
R7	None	None	
R8	BE14	20	±12
R9	BE11	458	±58
R10	BE12	178	±26
R11	BE8	2.625	±243

means that all repairs made in the model are triggered either by a component failing or by an inspection indicating that a repair needs to be executed.

4. RESULTS

The average value for the counters introduced in the inspection and the repair model can be found in Table 2 and Table 3 respectively. For I6, I7, R7 and R8, no values are stated. These modules are defined in the model but not used in the current version.
The average value of the counters introduced to the component models, and their associated inspection and repair modules can be seen in Table 4.
These numbers do not match exactly. The values marked with ± are the error bounds reported by Uppaal. For the large values, this is between 10−20%. For the small values, this error bound can be as high as 100%. This could be the result of running just a 1000 simulations. Increasing the number of simulations used to obtain these numbers, could decrease these error bounds.

The simulation for BE14 had to be re-run. The component was repaired more often than it should have been based on the inspections. This was caused by a typo in an identifier [Ruijters, personal correspondence]. A failure listener, responsible for noticing a component has failed and triggering a repair for that component, was listening to the top of the FMT, instead of to the component it was responsible for. This typo would have had no influence on the numbers of failures presented in the previous study. However, it could have had an impact on the cost of repairs used in the maintenance strategy analysis. It also had a drastic impact on the new counters, 1300 repairs in the first run compared to 20 repairs in the second run.

The most interesting value obtained from these simulations is the number of repairs made by repair module 2. Repair module 2 is responsible for replacing the entire joint, thus has the most impact on the expected failure number of the different components. In Table 3, it can be seen that the entire joint is replaced approximately 3600 times. This is more or less in line with the number of joints replaced indicated by ProRail, mentioned in 3.3: 3000. These 3600 repairs were caused by approximately 2000 inspections on BE2, approx. 1400 inspections on BE4 and approx. 10 inspections on BE15.
These 3600 repairs are also the difference between the total times a component is repaired and the number of times an inspection linked to the component triggers a repair, which is visible in Table 4.
BE12 is an exception: It indicates a joint being shorted because of the presence of a foreign object. Then it becomes obvious that repairing this failure is achieved by removing the foreign object, and nothing else[3]. It is also logical that replacing the entire joint has no influence on the number of times BE12 is repaired[4].

The results from Table 4 potentially point to an error in the model. This error involves the components that have their inspection threshold in the same phase as their failure moment. When such a component fails, it first notifies the failure listener that the component has failed, and that a repair should be performed as soon as possible. Then, notifies the inspection module that on the next scheduled inspection, it can detect degradation that should trigger a repair.
The failure listener will notify the repair module that it needs to perform a repair The repair module requires a period of time to perform the repair, in the current model this is one day. After this time has elapsed, it will repair the failed component, and any other components linked to this repair module.
Meanwhile, the inspection module is waiting for the next inspection, which happens once every 90 days. This means that it will take between 1 and 89 days before the next inspection takes place. Once this inspection takes place, it will notify the repair module that a repair should be done. Unless the inspection happens right after the failure, this means that the component was already repaired, and will be repaired again.
This could explain the difference between the number of inspection thresholds reached and the number of repairs made for BE11. BE11 should only require approximately 250 repairs, according to both the inspection rate and the number of failures predicted in Table 1, but is repaired approx. 450 times.

5. DISCUSSIONS

When comparing the number of times the components are repaired, as seen in Table 4, column 4, to the number of times the components are expected to fail, as seen in Table 1, column 6, it can be seen that the repairs either exceed or match the number of times the components where expected to fail. The only exception is BE9, but this is caused by the fact that BE9 was reported to have a proba-

[3] An exception would be if this foreign object caused damage when it made contact with the joint. However, such a failure is outside of the scope of this project.
[4] An exception would be a foreign object being left behind in the joint due to the maintenance. Again, this is outside the scope of this project.

Table 4. Comparison of the number of repairs performed on the basic events and the number of times the basic event reached the inspection threshold.

BE nr.	Inspected by	Repaired by repair modules	#Inspection threshold reached	#Maintenance repairs	#Failure repairs	#Repairs not replacing the joint	#Repairs replacing the joint
1	I1	R1, R2	1.145 ±227	4.647 ±383	115 ±30	1.363 ±237	3.669 ±330
2	I2	R2	1.973 ±191	3.644 ±321	117 ±22	0	3.669 ±330
3	None	R2	0	3.669 ±336	1 ±2	0	3.669 ±330
4	I2	R2	1.364 ±133	3.766 ±328	77 ±18	0	3.669 ±330
5	I5	R2, R5	1.063 ±113	4.646 ±430	53 ±14	1.247 ±122	3.669 ±330
6	I3	R2, R3	137 ±29	6.378 ±433	5 ±4	2.359 ±284	3.669 ±330
7	I3	R2	0[1]	3.761 ±334	11 ±15	0	3.669 ±330
8	I10	R2, R11	2.665 ±250	6.861 ±571	165 ±26	2.625 ±244	3.669 ±330
9	I3	R2, R3	2.124 ±257	5.699 ±408	194 ±37	2.359 ±284	3.669 ±330
10	I4	R2, R4	1.889 ±311	5.236 ±437	154 ±38	1.826 ±325	3.669 ±330
11	I9	R2, R9	240[1] ±30	4.012 ±336	238 ±31	458 ±59	3.669 ±330
12	None	R10	207[1] ±29	0	192 ±28	178 ±26	0
13	I4	R2, R4	4[1] ±3	5.822 ±482	9 ±5	1.826 ±325	3.669 ±330
14	I8	R2, R8	10[1] ±6	3.894 ±338	14 ±7	20 ±12	3.669 ±330
15	I2	R2	7[1] ±5	3.656 ±337	16 ±8	0	3.669 ±330

[1]This component reaches the inspection threshold at the same time that it fails

bility of 100% in Tabel 1, while BE9 only has a probability of 20% in the model.
The fact that the number of repairs matches the number of expected failures could indicate that the degradation is being modelled quite accurately. It is, however, necessary to compare the number of times the model says a component is repaired, to the number of times the component was actually repaired. This data was not available, unless the actual failures listed in the previous study also includes the repairs made due to inspection. However, this seems unlikely as those failures would not result in 3000 joints being replaced.
Analysing the components that share inspections and repairs indicated that BE6 and BE9 share an inspection and a repair module. This is of particular interest because no link is visible between these components in the initial FMT in Figure 3. of [6]. It is unclear if this was impossible to visualise in the FMT or if it is not supposed to be linked in the model.
Furthermore, BE7 also shares the inspection module with BE6 and BE9, but not the repair module. This means that the inspection module can find it is necessary to trigger a repair based on the degradation of BE7, but the repair module that is triggered will only repair BE6 and BE9. Of the approximately 2300 repairs triggered by the inspection module, approx. 2100 were triggered by BE9 and approx. 130 were triggered by BE6, while 0 inspections were triggered by BE7. Although this would indicate BE7 does not have an influence on the number of repairs, it does not indicate if BE7 should also be repaired, or if it should not be linked to this specific inspection module.
It was confirmed that BE7 should not share this inspection module, and should use the inspection module that triggers the joint replacement instead [Ruijters, personal correspondence].

With respect to the fact that inspections might trigger a repair when this is no longer necessary, as explained at the end of Section 4, this could be improved by changing the model. This can be achieved by creating an interaction between the repair module and the inspection module. If a repair module performs a repair, it should notify the inspection module that the inspection threshold is no longer reached.

5.1 Average values

All the values used in this paper, and the previous study, are averages. These values are obtained by modelling a single joint. The values from this joint are then multiplied by the number of joints that are present, in this case 50.000. Sometimes the result of this calculation is then divided by the duration of the simulation, to obtain the average number of failures in a single year.
It is not self-evident that creating such an average does justice to the different circumstances possible for joints.
A joint on a very busy railway will likely degrade faster than a joint on a railway that is barely used. Furthermore, the ground beneath a joint in Twente is far less likely to deform due to subsidence than the ground beneath a joint in the Randstad.
For circumstances that show a significant difference throughout the population, the distribution of joints affected could be calculated. This distribution could perhaps be modelled with a separate component that influences associated degradation using the RDEP-gate.
This model also disregards the effect of a joint failing on the probability of joints in close proximity also failing. It seems reasonable to assume that joints that are in close proximity to each other were installed at about the same time. Furthermore, they have experienced very similar conditions, thus one joint failing could indicate that other joints are also close to the end of their lifetime. If a joint is replaced, the railway section will need to be closed. The main cost of replacing the joint is probably closing the railway section and bringing the necessary machines to the location. It could be more cost efficient to replace the other joints that could fail soon, instead of waiting for them to fail and then closing the railway again. For repairs triggered by inspections, it seems reasonable to assume that this is in fact what happens. The inspection discovers degradations in multiple joints and they are repaired all at once. For repairs not triggered by inspections, it is harder to make assumptions.
Because the model works with a single joint, it is currently impossible to include such correlations. This could be solved by extending the current model to contain multiple copies of the joint. In such a model, it could be possible to have some of the joints share inspections and repairs.

6. CONCLUSIONS

To answer the main Research Question, it seems likely that it is possible to improve the model, but different improvements than initially expected.
Further answering the subquestions, it is possible to reproduce the results produced by the previous study, but those results were, perhaps, incomplete. The number of repairs was not counted. Analysing the number of repairs led to the discovery of at least one mistake, a typo. Another possible mistake involves the behaviour of components reaching an inspection threshold after they have already failed.
It has been shown that many repairs are performed in the simulation. Most of these repairs prevent failures that would have occurred. Every repair effectively removes an expected failure. If the model performs an unrealistic amount of repairs, this would create an unreliable prediction for the number of failures.

The results obtained in this study show a substantial variance. This is likely caused by the limited number of simulations that have been run to obtain these results. A higher number of simulations should be run to reduce this variance.

6.1 Future work

It has become clear that there are aspects of the model that require more verification. The number of repairs performed in the simulation needs to be compared to the number of repairs performed in the real world.

It might also be interesting to analyse the difference between maintenance and repair. In the current model, both actions bring a component to the first phase.
When maintaining an object, repairs often do not bring an object back to their initial state, this can only be accomplished by replacing the object. This could be modelled by replacements setting an object to the first phase, while repairs set it to the second phase, when this is possible and this also does not immediately trigger another repair.

Furthermore, to improve how degradation is modelled, it will likely be more effective to develop a new model that does not include maintenance. Different data would be necessary to verify the accuracy of such a model.
It might even be useful to perform real-world tests on the failure behaviour without maintenance. It will probably be expensive to create completely similar circumstances in such tests.
Ahktar et al.[1] created a test track and used artificial degradation to obtain information about the degradation of fishplates under different circumstances. However, the artificial degradation reduces the reliability of the obtained data.
A different option could be to place redundant joints in the railway, at an easily accessible location such as a train station. These joints would experience the same conditions as other joints but would not need the same amount of maintenance. Only degradation that endangers the integrity of the rail would need to be repaired.
If this leads to a dependable model, attempts could be made to change the current model to incorporate the way degradation should be modelled.

7. ACKNOWLEDGMENTS

We would like to thank Enno Ruijters and Carlos E. Budde for their constructive feedback and patience. We would also like to thank Sharon Vonk for her support and encouragement throughout the Bachelor.

8. REFERENCES

[1] M. Akhtar, D. Davis, L. Maal, J. Gordon, and D. Jeong. Effects of track parameters on rail joint bar stresses and crack growth. In *Arema 2010 Annual Conference and Exposition, Orlando, Florida*, 2010.

[2] G. Behrmann, A. David, and K. G. Larsen. A tutorial on uppaal 4.0, 2006.

[3] P. Bulychev, A. David, K. Larsen, M. Mikucionis, D. Poulsen, A. Legay, and Z. Wang. Uppaal-smc: Statistical model checking for priced timed automata. *EPTCS*, 85:1–16, 2012.

[4] A. David, K. Larsen, A. Legay, M. Mikučionis, and D. B. Poulsen. Uppaal smc tutorial. *Int J Softw Tools Technol Transfer*, 17:397, 2015.

[5] E. Ruijters, D. Guck, P. Drolenga, and M. Stoelinga. Fault maintenance trees: reliability centered maintenance via statistical model checking. In *Proceedings of the IEEE 62nd Annual Reliability and Maintainability Symposium (RAMS)*. IEEE, Jan. 2016.

[6] E. Ruijters, D. Guck, M. van Noort, and M. Stoelinga. Reliability-centered maintenance of the electrically insulated railway joint via fault tree analysis: A practical experience report. In *Proceedings of the 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 662–669. IEEE, 2016.

[7] W. Vesely, F. Goldberg, N. Roberts, and D. Haasl. Fault tree handbook. *Office of Nuclear Regulatory Research*, 1981.

[8] D. J. Wilkins. The bathtub curve and product failure behavior part one – the bathtub curve, infant mortality and burn-in. http://www.weibull.com/hotwire/issue21/hottopics21.htm. Accessed: 1-25-2018.

[9] D. J. Wilkins. The bathtub curve and product failure behavior part two – normal life and wear-out. http://www.weibull.com/hotwire/issue22/hottopics22.htm. Accessed: 1-25-2018.