Safety Analysis of the Height Control System for the Elbtunnel

Frank Ortmeier[1], Gerhard Schellhorn[1], Andreas Thums[1], Wolfgang Reif[1]
Bernhard Hering[2], and Helmut Trappschuh[2]

[1] Lehrstuhl für Softwaretechnik und Programmiersprachen,
Universität Augsburg, D-86135 Augsburg
{ortmeier,schellhorn,thums,reif}@informatik.uni-augsburg.de
[2] Siemens – I&S ITS IEC OS, D-81359 München
{bernhard.hering@atd.mchh,helmut.trappschuh@abgw}.siemens.de

Abstract. A new tunnel tube crossing the river Elbe has been built in Hamburg until the end of 2002. Therefore, a new height control system was required. A computer examines the signals from light barriers and overhead sensors to detect vehicles, which try to drive into a tube with insufficient height. If necessary, it raises an alarm that blocks the road. This paper describes the application of two safety analysis techniques on this embedded system: model checking has been used to prove functional correctness with respect to a formal model. Fault tree analysis has validated the model and considered technical defects. Their combination uncovered a safety flaw, led to a precise requirement specification for the software, and showed various ways to improve system safety.

Key words: safety analysis, fault tree analysis, formal methods, model checking

1 Introduction

This paper presents the safety analysis of the height control for the Elbtunnel. It is a joint project of the University of Augsburg with Siemens, department 'Industrial Solutions and Services' in Munich, which are sub-contractors in the Elbtunnel project, responsible for the traffic engineering.

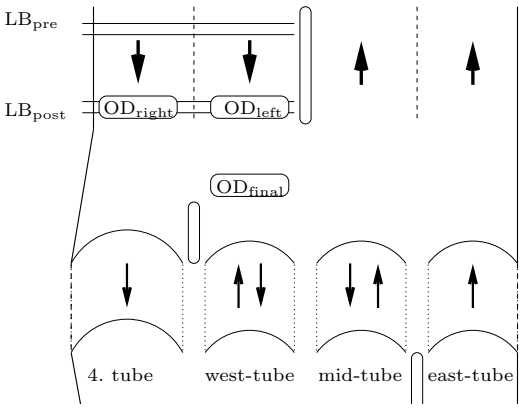
The Elbtunnel is located in Hamburg and goes beneath the river Elbe. Until December 2002 this tunnel had three tubes, where vehicles with a maximum height of 4 meters may drive through. A new, fourth tube has gone into operation in the year 2003. It is a larger tube and can be used by overhigh vehicles. A height control should prevent these overhigh vehicles from driving into the smaller tubes. It avoids collisions by triggering an emergency stop and locking the tunnel entrance.

Because the system consists of software and hardware components, we combine two orthogonal methods, model checking from the domain of software development and fault tree analysis from engineering. Model checking is used to prove safety properties like 'no collision'. Fault tree analysis (FTA) examines sensor failure and reliability.

We will briefly describe the layout of the tunnel, the location of the sensors, and its functionality in Sect. 2. Sect. 3 will introduce the methodology of the combined safety analysis approach. The formalization of the system and model checking of safety properties are presented in Sect. 4 and Sect. 5. The fault tree analysis in Sect. 6 completes the safety analysis. Some weaknesses of the system have been discovered which led to the proposals for improvements given in Sect. 7. Finally, Sect. 8 concludes.

2 The Elbtunnel Project

The Elbtunnel project is very complex. Besides building the tunnel tube, it contains traffic engineering aspects like dynamic route control, locking of tunnel tubes, etc. We will consider only a small part of the whole project, the height control. Before the fourth tube had been built, a height control existed for the 'old' three tubes. Light barriers were scanning the lanes for vehicles which are higher than 4 meters and triggered emergency stops. The existing height control had to be enhanced, such that it allows overhigh vehicles to drive through the new, higher tube, but not through the old ones.



HV	high vehicle
OHV	overhigh vehicle
LB	light barrier, detecting OHVs
OD	overhead detector, detecting HVs and OHVs (undistinguishable)
west-, mid-, east-tube	existing tubes; HVs may drive trough, but not OHVs
4. tube	new, higher tube; OHVs may drive trough

Fig. 1. Layout of the northern tunnel entrance

In the following, we will distinguish between *high vehicles* (HVs), which may drive through all tubes and *overhigh vehicles* (OHVs), which can only drive through the new, fourth tube. Figure 1 sketches the layout of the tunnel. The fourth tube may be cruised from north to south and the east-tube from south to north only. We focus our analysis on the northern entrance, because OHVs may only drive from north to south. The driving direction on each of the four lanes of the mid- and west-tube can be switched, depending on the traffic situation. Flexible barriers, signals and road fires guide drivers to the tubes, which are open in their direction.

The system uses two different types of sensors. Light barriers (LB) are scanning all lanes of one direction to detect, if an OHV passes. For technical reasons

they cannot be installed in such a way, that they only supervise one lane. Therefore overhead detectors (OD) are necessary to detect, on which lane a HV passes. ODs can distinguish vehicles (e.g. cars) from high vehicles (e.g. buses, trucks), but not HVs from OHVs (but light barriers can!). If the height control detects an OHV heading towards a different than the fourth tube, then an emergency stop is signaled, locking the tunnel entrance.

The idea of the height control is, that the detection starts, if an OHV drives through the light barrier LB_{pre}. To prevent unnecessary alarms through faulty triggering of LB_{pre}, the detection will be switched off after expiration of a timer (30 minutes). Road traffic regulations require, that after LB_{pre} both HVs and OHVs have to drive on the right lane through tunnel 4. If nevertheless an OHV drives on the left lane towards the west-tube, detected trough the combination of LB_{post} and OD_{left}, an emergency stop is triggered. If the OHV drives on the right lane through LB_{post}, it is still possible for the driver to switch to the left lanes and drive to the west- or mid-tube. To detect this situation, the height control uses the OD_{final} detector. To minimize undesired alarms (remember, that normal HVs may also trigger the ODs), a second timer will switch off detection at OD_{final} after 30 minutes. For safe operation it is necessary, that after the location of OD_{final} it is impossible to switch lanes. Infrequently, more than one OHV drives on the route. Therefore the height control keeps track of several but at the most three OHVs.

3 Methodology

In this section we describe our approach of the combined application of fault tree analysis and formal methods for analyzing system safety. We describe four phases of development. The first phase is used to get a detailed understanding of the application domain. The second phase consists of the safety analysis of the system with formal methods and fault tree analysis. These two analysis methods are applied separately, to preserve the different points of view – the functional and the safety view – of the system. The third phase serves for exchanging the outcome of the analysis which results in improvements of the formal model and the fault tree analysis. Finally, the fourth phase sums up the results.

Requirements Analysis (I) The aim of the first phase is to get a precise definition of the system requirements and the system boundary. This requires developers and customers to get a common understanding of the system functionality.

Semi-formal specifications (e.g. UML diagrams) and rapid prototyping can help to achieve this goal. They can also serve as a starting point for safety analysis, as depicted in figure 2.

In our case, where the authors from university were not familiar with the application domain of traffic engineering, we decided to build an executable model of the height control. We used the *Statemate* tool [4] which supports simulation of statechart models.

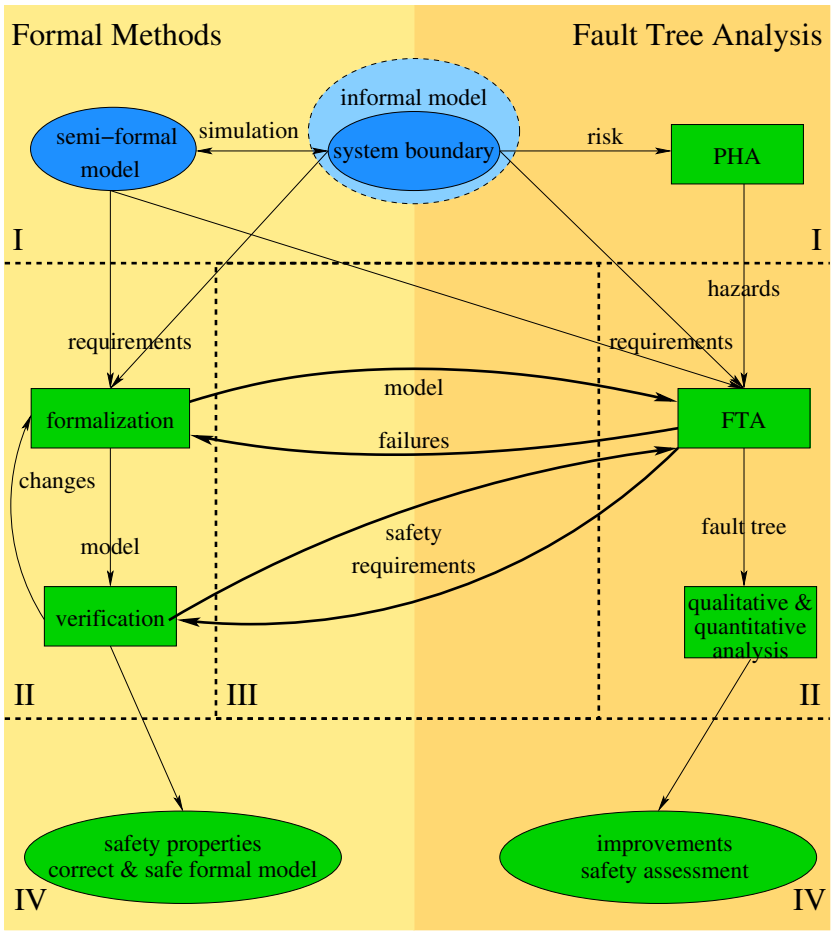


Fig. 2. Process Model

The control part of the height control is modeled with statecharts, while sensor signals are input events that can be controlled via a graphical user interface. Simulating the system enabled us to discuss and understand the interaction of the different sensors and the required reaction of the height control in detail.

The informal system description, the Statemate model, and various discussions resulted in detailed requirement definitions for the height control of the Elbtunnel.

Safety Analysis (II) The second phase starts with the independent development of FTA and formal analysis by model checking. Our experiences show, that it is very important to develop the formal model and the fault tree independently. Otherwise, the FTA is focused too much on the formal system

description and looses the view on unspecified behaviors, as possible defects and failures of system components. A precondition for this independent approach is the same understanding of the system and the system boundaries, which are exactly defined in the first phase.

For the formal treatment of the system, the informal requirements have to be formalized, resulting in a formal system model. Because we would like to prove safety properties automatically by model checking, a finite automata model was developed, which is described in Sect. 4. Validation against the informal description and the Statemate specification improves the formal model. This model is the basis for the verification of the theorems mentioned in Sect. 5.

For fault tree analysis, a list of overall system hazards or undesired events has to be developed. This is a result of the preliminary hazard analysis (PHA) [5]. In the example, the critical events are 'collision' and 'undesired emergency stops'. They are analyzed further with FTA. Fault tree analysis systematically breaks down undesired system events to defects and failures of components and examines, which combinations of component failures causes the hazard. Critical system components are discovered (see Sect. 6).

Integration (III) The third phase brings together the results of the formal and the safety analysis. The fault tree has to be adapted, to reflect the constraints of the formal model. We observed, that mostly not every cause-consequence relation in the fault tree respects these constraints. The other way round, component failures detected with FTA influence the formal model. Safety requirements are exchanged in both ways leading to changes in the formal model and the fault tree analysis. Verification and safety analysis has to be (partly) redone.

In this case study, we did this adaption in a very informal way by exchanging results from the different techniques and checking the consistency of the cause-consequence relations in the fault tree manually. Currently we are working on a more formal integration, where events in the fault tree are formalized and the cause-consequence relations are formally proven [10]. Nevertheless, we could achieve mutual benefits for the two analysis methods. The formal approach detected a flaw in the height control, which could lead to a collision, when two OHVs are passing simultaneously LB_{pre} (see Sect. 5 for details). This exceptional event was (at first) not mentioned in the FTA and can (in our opinion) only be found through rigorous reasoning. On the other hand, FTA uncovers additional safety requirements and failures. E.g., the control system has to consider situations, where tube four is not available or only tube four is available. These cases were not considered in the formal model at first, but were added and proved in the third phase (see theorem 3 in Sect. 5).

Results (IV) Finally, the fourth phase summarizes the results of both techniques. The overall process results in a correct and safe formal model with verified safety properties. In addition to the safety statement for an operational system, the safety analysis assesses the systems safety with respect to component failures and assists in detecting system improvements (see Sect. 7 for details). We think

that the presented approach is very useful for developing high assurance system specification.

4 Formal Specification

In this section we define a formal specification of the Elbtunnel using timed automata over a finite set of states. An automaton is shown as a directed graph, where states are nodes. A transition is visualized as an arrow marked with a condition c and a time interval [t1,t2] as shown in Fig. 3. A transition from s1 to s2 may be taken indeterministically at any time between t1 and t2, if the condition c holds from now until this time. The time interval is often just [t,t] (deterministic case) and we abbreviate this to t. If t = 1, a transition will happen in the next step (provided the condition holds) and we have the behavior of an ordinary, untimed automaton. The always true condition is indicated as '−'.

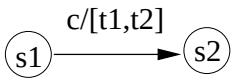


Fig. 3. Transition

The automata defined below are graphic representations of the textual specification that we used in the model checker RAVEN [8], [9], [7]. We also tried the model checker SMV [6], which supports untimed automata only, but is more efficient. Timed automata are translated to untimed ones using intermediate states which say "the system has been in state s1 for n steps", see [7].

The specification consists of two parts. The first specifies the control system that is realized in software and hardware, the second describes the environment, i.e. which possible routes HVs and OHVs can follow. Our aim is to prove (Sect. 5), that the control system correctly reacts to the environment: for example we will prove that if an OHV tries to drive into mid- or west-tube (behavior of the environment) then the control system will go into a state that signals an emergency stop (reaction of the control system).

Both parts of the specification consist of several automata: the control system consists of two automata CO_{pre} and CO_{post}, which will be implemented in software. The first counts OHVs between LB_{pre} and LB_{post}, the second checks if there are any after LB_{post}. Each uses a timer (TI_{pre} and TI_{post}), modeled as an instance of the same automaton with different input and output signals.

The environment consists of three identical automata OHV_1, OHV_2, OHV_3 for OHVs. This is sufficient, since it is assumed, that at most three OHVs may pass simultaneously through the tubes. Finally, three automata HV_{left}, HV_{right}, HV_{final} model HVs that trigger the sensors OD_{left}, OD_{right}, and OD_{final}. They are instances of a generic automaton describing HVs. Altogether the system consists of 10 automata running in parallel:

$$\begin{aligned} SYS = {} & CO_{pre} \parallel CO_{post} \parallel TI_{pre} \parallel TI_{post} \parallel \\ & OHV_1 \parallel OHV_2 \parallel OHV_3 \parallel HV_{left} \parallel HV_{right} \parallel HV_{final} \end{aligned}$$

The following sections will describe each automaton in detail.

Specification of a Timer The generic specification of a timer is shown in Fig. 4. Initially the timer is in state 'off', marked with the in-going arrow. When it receives a 'start' signal, it starts 'running' for time 'runtime'. During this time, another 'start' signal is interpreted as reset. After time 'runtime' the timer signals 'alarm' and finally turns off again. Two timers, which have a runtime of 30 minutes, are used in the control system. The first (TI_{pre}) is started when an OHV passes through LB_{pre}, i.e. 'start' is instantiated with LB_{pre}. After 30 minutes it signals TI_{pre}.alarm to CO_{pre}. The second timer TI_{post} is triggered by an OHV passing LB_{post}, but only if LB_{pre} has detected an OHV in the preceding 30 minutes (to avoid false alarms). As described in the next section, this is equivalent to the condition $CO_{pre} \neq 0$ and 'start' is instantiated with $LB_{post} \wedge CO_{pre} \neq 0$. The second timer signals TI_{post}.alarm to CO_{post}.

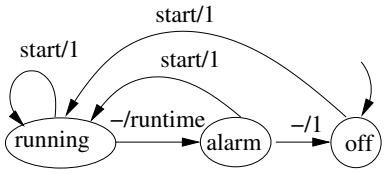


Fig. 4. Timer TI

Specification of Control for LB_{pre} The control automaton CO_{pre} (shown in Fig. 5) controls the number of OHVs between the two light barriers LB_{pre} and LB_{post}. Starting with a count of 0, every OHV passing LB_{pre} increments the counter, every OHV passing LB_{post} decrements it. If CO_{pre} receives TI_{pre}.alarm, i.e. if for 30 minutes no OHV has passed through LB_{post}, the counter is reset. Actually the automaton shown in Fig. 5 is not completely given, all edges which correspond to simultaneous events are left out for better readability.

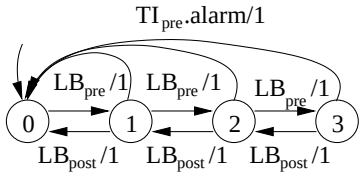


Fig. 5. Control CO_{pre} for LB_{pre}

Specification of Control for LB_{post} Figure 6 shows the automaton CO_{post} which controls OHVs in the area after LB_{post}. It has fewer states than CO_{pre}, since it just needs to know whether there is at least one OHV in the critical section between LB_{post} and the entries of the tubes, but not how many. If at least one OHV is in the critical section then the automaton is in state 'active', otherwise in 'free'. To signal an emergency stop, the automaton goes to state 'stop'. To avoid false alarms, CO_{post} interprets the interruption of LB_{post} as a misdetection, if CO_{pre} has not detected the OHV before (i.e. when $CO_{pre} = 0$). There are two reasons for an

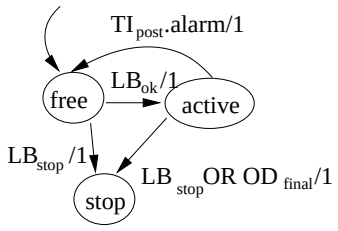


Fig. 6. Control CO_{post} for LB_{post}

emergency stop: either OD_{final} signals that an OHV (or a HV) tries to enter the mid- or west-tube, while an OHV is in the critical section (i.e. we are in state 'active'). Or an OHV tries to drive through LB_{post}, but has not obeyed the signs to drive on the right lane. This rule must be refined with two exceptions. If tube 4 is not available, an emergency stop must be caused even if the OHV is on the right lane. The other way round, an empergency stop must not be signalled, if only tube 4 is available, even if the OHV does not drive on the right lane. This means that the conditions LB_{stop} and LB_{ok} for going to the 'stop' resp. 'active' state must be defined as:

$$\begin{aligned} LB_{ok} \quad &:= \quad LB_{post} \text{ and } CO_{pre} \neq 0 \\ &\quad \text{and (only\_tube\_4\_open or } (OD_{right} \text{ and not } OD_{left})) \\ LB_{stop} &:= \quad LB_{post} \text{ and } CO_{pre} \neq 0 \\ &\quad \text{and (tube\_4\_closed or not } (OD_{right} \text{ and not } OD_{left})) \end{aligned}$$

Specification of Overhigh Vehicles The specification given in Fig. 7 shows the possible behavior of one OHV that drives through the Elbtunnel. It is the core of the environment specification. We can implement the control system, such that it behaves exactly as the specification prescribes but we have no such influence on reality. Whether the real environment behaves according to our model, must be validated carefully (e.g. with fault tree analysis, see Sect. 6).

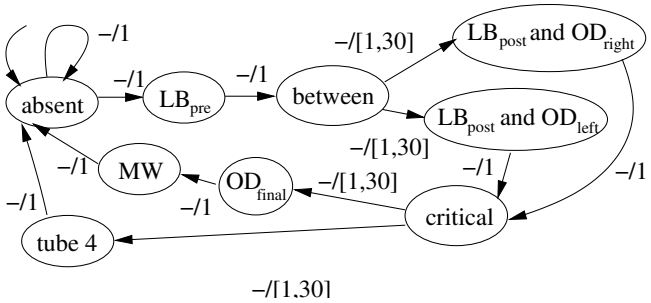


Fig. 7. Overhigh vehicle OHV

The model describes the possible movement of an OHV through the light barriers and the tunnel: Initially the OHV is 'absent'. It remains there for an unknown period of time (idling transition of 'absent' state). At any time it may reach LB_{pre}. In the next step the OHV will be 'between' the two light barriers and will remain there for up to 30 minutes. Then it will drive through LB_{post}. Either it will do so on the right lane, which will trigger OD_{right} or on the left lane and cause signal OD_{left}. After passing LB_{post} it will reach the 'critical' section where the lanes split into the lane to tube 4 and the lanes to mid- and west-tube. The OHV may stay in this critical section for up to 30 minutes. Then it will

either drive correctly to the entry of 'tube 4' and finally return to state 'absent' or it will pass through OD_{final} and reach the entry of the mid- or west-tube (state 'MW'). In this case the control system must be designed such that an emergency stop has been signalled and we assume that the OHV will then be towed away (return to state 'absent').

Our complete specification runs three automata OHV_1, OHV_2, OHV_3 in parallel. The signal LB_{pre}, that is sent to CO_{pre}, is the disjunction of the OHV_i being in state LB_{pre}. Signals LB_{post}, OD_{left} etc. are computed analogously.

Specification of High Vehicles High vehicles which are not overhigh are relevant for the formal specification only since they can trigger the ODs. Therefore it is not necessary to specify which possible routes they drive or how many there potentially may be. Instead we only define three automata OD_{left}, OD_{right} and OD_{final}, which say, that at any time the corresponding 'OD' signal may be triggered. Fig. 8 gives a generic automaton that randomly switches between the two states.

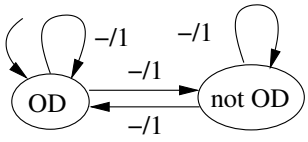


Fig. 8. High vehicle HV

5 Proving Safety Properties

The formal specification of the Elbtunnel allows to state safety properties and to prove them rigorously as theorems over the specification. Since the model has only finitely many states, we can use model checkers, which are able to prove theorems automatically. We used two model checkers, SMV and RAVEN. Both have the ability to generate counter examples, i.e. if a theorem we try to prove turns out to be wrong, they return a run of the system which is a counter example to the theorem.

To keep the number of states in the model small, we have also avoided an exact definition of the duration of one step of the automaton in real time. Of course the value of 30 (minutes), we used for the maximal time that an OHV is between the two LBs, is much more than 30 times longer than the time the OHV needs to cross an LB. For the real experiments we have even used a value of 5 or 6 to keep the runtimes of the proofs short (with a value of 30 proofs take several hours, while proofs using 5 go through in seconds). The exact value used in the proofs does not influence the results, as long as the maximum driving times and the runtime of the timers agree.

Despite these inadequacies the model can serve to analyze all safety properties we found relevant. The most important is of course, that whenever an OHV is driving to the entry of the mid- or west-tube, then an emergency stop will be raised. This is formalized by the following formula in temporal logic:

$$\mathrm{AG}((\mathrm{OHV}_1 = \mathrm{MW} \vee \mathrm{OHV}_2 = \mathrm{MW} \vee \mathrm{OHV}_3 = \mathrm{MW}) \rightarrow \mathrm{CO}_{\mathrm{post}} = \mathrm{stop})$$

The formula is read as: For every possible run of the system it is always the case (temporal operator AG), that if one of OHV_1, OHV_2, OHV_3 is in state 'MW', then CO_{post} is in state 'stop'.

The first verification results have been negative due to several specification errors. There have been basically two classes: simple typing errors, and errors due to simultaneous signals. E.g. our initial model included only transitions for one signal in the control automaton shown in Fig. 5 and left out the case that one OHV passes LB_{pre} while another passes LB_{post} at the same time. Additional transitions are necessary when two signals occur simultaneously (but they are left out in Fig. 5 for better readability). Both types of errors have easily been corrected, since each failed proof attempt resulted in a trace, that clearly showed what went wrong.

After we corrected the errors, we finally found, that we still could not prove the theorem. It does not hold as can be demonstrated by the following run:

1. Initially, there are no OHVs in the controlled section. This means all OHVs are 'absent', CO_{pre} is in state '0', and CO_{post} signals 'free'.
2. Then two OHVs drive through the light barrier LB_{pre} *at the same time*.
3. LB_{pre} cannot detect this situation, so CO_{pre} counts only one OHV and switches into state '1'. This means we now have two OHVs in the controlled section, but the system is only aware of *one*.
4. The first of the two OHVs travels the distance to LB_{post} in a short time and triggers LB_{post}. This resets the state of CO_{pre} to '0', switches CO_{post} into the 'active' state, and starts timer TI_{post}.
5. The other OHV takes some time to reach LB_{post} (possibly a traffic jam on its lane). When this OHV triggers the light barrier, CO_{post} assumes the signal from LB_{post} to be a misdetection, since CO_{pre} is in state '0'. Therefore it does not restart timer TI_{post}.
6. The second OHV now needs longer than the remaining time of TI_{post} to reach OD_{final}, remember TI_{post} is already running for quite some time. TI_{post} elapses and signals CO_{post} to switch to state 'free' again.
7. The OHV triggering OD_{final} while CO_{post} is in state 'free' is assumed to be a misdetection and does not cause an emergency stop.
8. Finally, the OHV can now enter the mid- or west-tube without having caused an emergency stop.

This run shows that our system is inherently unsafe, since two OHVs that pass simultaneously through LB_{pre} are recognized as one. Whether the flaw is relevant in practice, depends on the probability of its occurrence, which is currently being analyzed.

To get a provable safety property, we now have two possibilities: either we can modify the model, possible modifications for which the safety property can be proved are discussed in Sect. 7. Or we can weaken the safety property to exclude this critical scenario. Then we can prove that the critical scenario described above is the *only* reason that may cause safety to be violated:

Theorem 1. *safety*
System SYS has the following property: if two OHVs never pass simultaneously through LB_{pre}, then any OHV trying to enter the middle or western tube will cause an emergency stop.

In practice this means, that if the height control is implemented following the description of Sect. 4, then it will be safe except for the safety flaw described above. It is interesting to note, that we detected the flaw only in the RAVEN specification. We did not find the problem with SMV, since we had made an error in the translation of timed automata to untimed automata. This error resulted in a specification in which all OHVs had to stay for the full 30 minutes between LB_{pre} and LB_{post}. This prevented the problem from showing up: two OHVs driving simultaneously through LB_{pre} also had to drive through LB_{post} at the same time. The incident shows that using a specification language which does not explicitly support time is a source for additional specification errors.

Safety is not the only important property of height control in the Elbtunnel (although the most critical). Another desirable property is the absence of unnecessary alarms. Here we could prove:

Theorem 2. *emergency stops*
If tube 4 is open and not the only open tube, then an emergency stop can only be caused by

- an OHV driving on the left lane through LB_{post} or
- an OHV driving through OD_{final} or
- an OHV driving on the right lane through LB_{post} while a high vehicle is driving on the left lane or
- a high vehicle at OD_{final} while Timer TI_{post} is running.

The first two causes are correctly detected alarms, the other two are false alarms inherent in the technical realization of the system (and of course already known). Formal verification proves the absence of other causes for false alarms.

Finally, we analyzed, how the availability of tubes influences the situation. This led to the following theorem:

Theorem 3. *tube 4 availability*
If tube 4 is not available, then any OHV trying to drive through will cause an emergency stop. If only tube 4 is available, then an emergency stop will never occur.

Summarizing, the formal analysis has led to a precise requirement specification for the automata that should be used in the height control. A safety flaw has been discovered and proven to be the only risk for safety. False alarms due to other reasons than the ones mentioned in Theorem 2 have been ruled out. The results of formal analysis show that the control system does not have inherent logic faults.

The formal proofs of this section do not give an absolute safety guarantee, but only relative to the aspects considered in the formal model. For example we did not consider technical defects, which are covered by the fault tree analysis presented in the next section.

6 Fault Tree Analysis

Another approach to increase overall system safety is fault tree analysis (FTA)[11]. FTA is a technique for analyzing the possible basic causes (primary failures) for a given hazard (top event). The top event is always the root of the fault tree and primary failures are its leaves. All inner nodes of the tree are called intermediate events (see Fig. 9). Starting with the top event the tree is generated by determining the immediate causes that lead to the top event. They are connected to their consequence through a gate. The gate indicates if all (and-gate) or any (or-gate) of the causes are necessary to make the consequence happen. This procedure has to be applied recursively to all causes until the desired level of granularity is reached (this means all causes are primary failures that won't be investigated further).

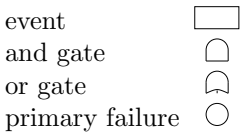


Fig. 9. FT Symbols

We analyzed two different hazards for the Elbtunnel height control - the collision of an OHV with the tunnel entrance and the tripping of a false alarm. We will use the hazard collision to illustrate FTA (see Fig. 10).

The immediate causes of the top event - collision of an OHV with tunnel entrance - are that either the driver ignores the stop signals OR (this means the causes are connected through an OR-gate) that the signals are not turned on. The first cause is a primary failure. We can do nothing about it, but to disbar the driver from his license. The second cause is an intermediate event. It's immediate causes are a) that the signal lights are out of order or b) the signals were not activated. Again the first one is a primary failure and the second is an intermediate event, which has to be investigated further. We will not present the whole tree here, but only discuss the most interesting results. One of these is the fact, that the original control system had a safety gap.

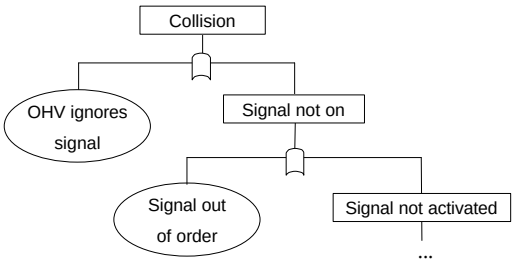


Fig. 10. Fault tree for hazard collision

This gap may be seen in one of the branches of the fault tree (see Fig. 11). The direct causes for the (intermediate) event OHV not detected at LB_{pre},

are malfunctioning of LB_{pre} or synchronous passing of two OHVs through the light barrier LB_{pre}. The malfunction is a primary failure. But the second cause represents a safety gap in system design. In contrast to all other primary failures in the fault tree this event is a legal scenario. Although all components of the system are working according to their specification, the hazard collision may still occur. This must not happen in a safe control system.

Another way to use FTA is to examine the systems sensibility to component failures. When analyzing the fault tree for the collision hazard, we found, that there are no AND-gates in the collision fault tree. This means that there is no redundancy in the system - making it effective on the one hand but susceptible to failure of each component on the other hand (with respect to collisions).

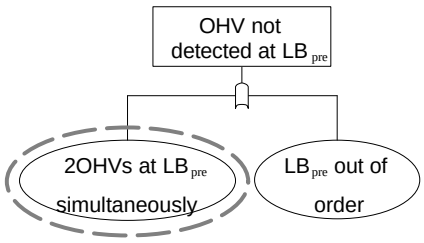


Fig. 11. Safety gap

The false alarm fault tree is different. This tree has several AND-gates. Especially, misdetection of pre-control light barrier LB_{pre} appears in almost all minimal cut-sets (at least in all those scenarios where no OHV is involved at all). This means that the system is resistant to single point failures (i.e. only one primary failure occurs) with regard to the triggering of false alarms. Most failure modes in this fault tree are misdetections. These failures are by far more probable than a not detected OHV, e.g. a light barrier can easily interpret the interruption by a passing bird as an OHV but it is very improbable that it still detects the light beam while an OHV is passing through.

An important result of fault tree analysis are the minimal cutsets. A cutset is a set of primary failures, which are necessary to make the hazard happen for sure. A minimal cutset is a cutset, such that preventing one of its primary failures prevents the hazard from occuring. Cutsets which consist of only one element are called single point failures. This means the system is very susceptible to this primary failure. The probability of a minimal cutset is, assuming statistical independence, the product of all its primary failure probabilities. The probability of the hazard can be computed by summing up all minimal cutsets probabilites. This is a standard technique in engineering and is called quantitative fault tree analysis [11].

Using this technique we could show, that all the measures taken are complementary in their effects on the two discussed main hazards, collision and false alarm, as shown in Fig. 12. E.g. the pre-control LB_{pre} decreases the risk of false alarms, but - less obvious - it increases the risk for collisions as well. This is because a misdetection of LB_{pre} is a primary failure in the collision fault tree. On the other hand the failure mode "faulty detection at LB_{pre}" is part of every minimal cutset of the false alarm fault tree. In our current analysis we do not use real statistical data, as they are not accessible to us yet. But we can still give

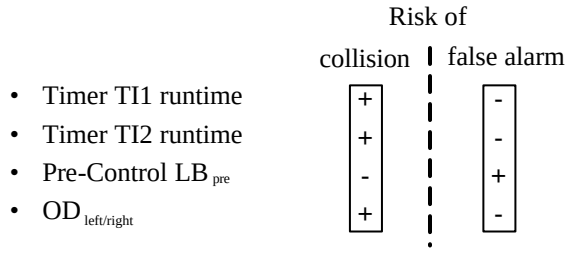


Fig. 12. Complementary effects

qualitative advice using qualitative information on the failure probabilities (e.g. the probability of a misdection of an overhead detector is far greater than that of a light barrier). Using this information we may show properties like $\mathrm{LB}_{\mathrm{pre}}$ decreases the risk of false alarms a lot, while increasing the risk of collisions only insignificantly. This result correlates with the intention of the system to decrease the high number of false alarms significantly, while still keeping the height control safe in terms of collision detection (the actual height control triggers about 700 false alarms per year). The development of a more detailed technique, which allows to compare different system designs and yields optimal configuration values for the different parameters, like the runtimes of the timers in this case study, is a current research project of our group.

7 Improvements

The safety analysis led to some suggestions for improvements and changes in the control logic as well as in the physical placement of the sensors. These changes were handed back as analysis feedback to our partners. Their cost-value benefit is being discussed and it is very likely that they will be implemented. In the first part of this section we describe two possible solutions to close the safety leak. Then we will explain some improvements for the overall performance and quality of the system, which were discovered through the safety analysis.

Measures to Close the Safety Gap The first suggestion is the better one in terms of failure probability, but the second one can be implemented without any additional costs.

Installing additional ODs at LB_{pre}. The first possibility of closing the safety gap of simultaneous passing of 2 OHVs at $\mathrm{LB}_{\mathrm{pre}}$ is to install additional ODs at the pre-control signal bridge. With ODs above each lane one can keep track of the actual number of HVs that are simultaneously passing the pre-control checkpoint. Every time the light barrier is triggered, the counter $\mathrm{CO}_{\mathrm{pre}}$ (see Fig. 5) will not only be increased by one but by the number of HVs detected by the (new) ODs. With this information it may be assured that the counter is always at least equal to the number of OHVs in the area between pre- and

post-control. The counter can still be incorrect, e.g. simultaneous passing of an OHV and a HV through the pre-control LB_{pre} will increase it by two (instead of one), but it can only be greater than the actual number of OHVs. This closes the safety gap. It increases the probability of false alarms only insignificantly.

Never stop TI_{pre}. An alternative to the solution described above is to keep TI_{pre} always active for its maximum running time and restart it with every new OHV entering the controlled area. If this solution is chosen, the counter CO_{pre} will be redundant; it can be replaced by an automaton similar to CO_{post}. The advantage of this measure is of course that no additional sensors must be installed. Only a change in the control logic is required. On the other hand it has a higher increase in false alarm probability than the option with ODs (as the alarm triggering detectors are kept active longer).

Changes to Improve Overall System Quality We will now give useful advice for changes, which increase the overall performance and quality of the system. As already mentioned, quantitative information on failure probabilites and hazard cost were not available to us. But the presented changes will improve the overall quality for realistic failure rates. Both measures described here are aimed at reducing the number of false alarms.

Additional light barrier at entrance of tube 4. The FTA showed that one important factor for the hazard false alarm is the total activation time of OD_{final}. This is because each HV passing one of the OD_{final} sensors immediately leads to a false alarm, if the sensor is activated. As described above, these detectors are active while TI_{post} is running. An additional light barrier at the entrance of tube 4 can be used to detect OHVs that are leaving the endangered area between post control and tunnel entrance. This can be used to stop TI_{post} and keep it only running while there are OHVs left in the last section. It will be necessary to use a counter to keep track of the number of OHVs in the post sector. Another advantage is that the timeout for TI_{post} may be chosen much more conservatively without significantly increasing the risk of false alarms, but decreasing the risk of collisions a lot. It is important to make the risk of misdetections of this additional light barrier as low as possible as misdetections could immediately lead to collisions. This can be done by installing a pair of light barriers instead of a single one and connecting them with an AND-connector.

Distinguished alarms at post control. To further decrease the risk of false alarm, one may only trigger an alarm at post control if OD_{left} detects one HV. If neither OD_{left} nor OD_{right} detect a HV (or OHV), OD_{final} will be activated without triggering an alarm. This means the system assumes that the post control light barrier had a misdetection, if both ODs can't detect a high vehicle. But it still activates OD_{final} just in case (if either OD_{left} or OD_{right} are defect). This measure increases the risk of collisions almost unnoticeable. But the reaction time between alarm signals and the potential collision will decrease.

8 Conclusion

The safety analysis of the height control for the Elbtunnel has shown the benefit of combining formal verification and fault tree analysis. Both, formal model checking and safety analysis are important to examine a complex system. Formal verification gives precise requirements specification for the control system and discovers logical flaws in the control software. In our case study it has revealed the safety problem of two OHVs passing LB_{pre} at the same time.

On the other hand, building an adequate model for the system and its environment is not a simple process. To minimize specification errors an orthogonal analysis technique like FTA is needed. FTA also addresses the issue of component failures. This analysis gives a lot of useful hints for improving the system. The revised system can be formally model checked again leading to a synergy effect.

The presented case study, which required an effort of 3 person months, combines both techniques by exchanging results. For a tighter integration a FTA semantics was developed [10] used to formally verify the completeness of fault trees. Proof support [2] is integrated into the interactive specification and verification system KIV [1] using statecharts [3] as formal modeling language.

In conclusion, we find that the combination of formal methods and FTA is a suitable analysis technique for embedded systems. Our analysis has made the system safer (by detecting and closing the safety gap), led to design improvements and increased overall system quality.

References

[1] M. Balser, W. Reif, G. Schellhorn, K. Stenzel, and A. Thums. Formal system development with KIV. In T. Maibaum, editor, *Fundamental Approaches to Software Engineering*, number 1783 in LNCS. Springer, 2000.

[2] M. Balser and A. Thums. Interactive verification of statecharts. In *Integration of Software Specification Techniques (INT'02)*, 2002.

[3] D. Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8(3), 1987.

[4] D. Harel, H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. Shtull-Trauring, and M. Trakhtenbrot. Statemate: A working environment for the development of complex reactive systems. *IEEE Transactions on Software Engineering*, 16(4), 1990.

[5] N. Leveson. *Safeware: System Safety and Computers.* Addison Wesley, 1995.

[6] K. L. McMillan. *Symbolic Model Checking.* Kluwer Academic Publishers, 1990.

[7] J. Ruf. RAVEN: Real-time analyzing and verification environment. Technical Report WSI 2000-3, University of Tübingen, Wilhelm-Schickard-Institute, January 2000.

[8] Jürgen Ruf and Thomas Kropf. Symbolic Model Checking for a Discrete Clocked Temporal Logic with Intervals. In E. Cerny and D.K. Probst, editors, *Conference on Correct Hardware Design and Verification Methods (CHARME)*, pages 146–166, Montreal, 1997. IFIP WG 10.5, Chapman and Hall.

[9] Jürgen Ruf and Thomas Kropf. Modeling and Checking Networks of Communicating Real-Time Systems. In *Correct Hardware Design and Verification Methods (CHARME 99)*, pages 265–279. IFIP WG 10.5, Springer, September 1999.

[10] G. Schellhorn, A. Thums, and W. Reif. Formal fault tree semantics. In *Proceedings of The Sixth World Conference on Integrated Design & Process Technology*, Pasadena, CA, 2002.

[11] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl. *Fault Tree Handbook*. Washington, D.C., 1981. NUREG-0492.