Formal model for generating railway interlocking software based on a modularized track model

DANIEL TÖPEL, JOSEF BÖRCSÖK
University Kassel
Computer Architecture and System Programming
Wilhelmshöher Allee 71, 34121 Kassel
GERMANY
dtoepel@uni-kassel.de, j.boercsoek@uni-kassel.de

Abstract: This paper discusses the possibility to create railway interlocking software in an automated process. The classic approaches are interlocking tables or relay modules. In the case of using interlocking tables, they form the core of the signal box. This means ideally that two signal boxes differ only by their interlocking tables. The major disadvantage of interlocking tables is the massive manual work to create and verify them. The paper will first define its own railway regulations which are a subset of the German train regulations. It will then formally design a model for a blank signal box. The next step is the customization of the empty signal box. This will be done by a graphical track editor which enables a user to enter a track layout and creates the interlocking table from the topology.

Key–Words: Interlocking, Railway, Safety, Verification, Train control

1 Introduction

Early signal boxes were built up on a mechanical logic. A locking mechanism, which had to be designed manually, was installed to prevent dangerous train movements. An interlocking table was created from the track layout. Once the interlocking table was considered to be correct, it could be implemented as a combination of locking bars and tappets [1].

Another concept was represented by relay signal boxes [3]. Each element of infrastructure was symbolized by a module. These modules were wired in the signal box according to their topology. A locking mechanism was no longer required, as decisions were made means of electric circuits running through the involved modules.

Modern electronic signal boxes returned to the concept of locking mechanisms [3]. Without the restrictions of a physical mechanism, electronic signal boxes are more flexible and able to handle larger layouts than mechanical signal boxes.

This paper is intended to investigate on the possibility to create the software for a signal box from a track editor/designer software. The user should be able to place tracks, signals, points and other infrastructure into his model. On the basis of these data the software for a central unit, infrastructure and a visualisation for the train dispatcher shall be created.

The creation of interlocking tables has been occasionally investigated. An approach on creating tables from a topology for Turkish regulations has been discussed in [2]. This paper will be based on German regulations instead and will additionally consider ‘early partial route release’ as a measure to enhance the capacity of the line.

This paper is organized the following way. Section 2 will describe the hardware and software, which is available to design the signal box. In section 3 the rules of the railway are defined, according to which the data structure and behaviour for the signal box is modelled in section 4. This model will be tested in section 5 with an example. Section 6 discusses the results and a perspective will be shown in section 7.

2 Hardware, Software

The central computing unit will be running on a safety related system to ensure the required safety level, defined by EN 50126 [4], EN 50128 [5], EN 50129 [6] and IEC 61508 [7]. The infrastructure will also run their safety related systems. In this approach those will not be considered. The complete signal box software will consist of the following subunits.

- Central computing unit
- Track Editor
- Dispatcher’s view
- Infrastructure communication units
- Safe communication layer
2.1 Track Editor

The track editor (see Fig. 1) is used to design the track layout. The program checks for inconsistencies and generates data for the other parts of the software. The track editor does not run once the signal box is running. It is written in Java, as the program is not required to run safely. Nevertheless, it is required to prove that the program produces only correct results.

2.2 Central Computing Unit

The Central Computing Unit (CU) is gathering information from the infrastructure and receiving orders from the dispatcher. It will then make a decision and distribute the orders to the infrastructure. This program is to be run under a safe environment. The CU should avoid any unnecessary complexity to ease verification. In this approach it will have to solve the following tasks.

- Store information about the current state of the infrastructure.
- Manage a list of active train movements.
- Make decisions about allowing new train movements.
- React to requests from other parts of the signal box.

2.3 Dispatcher’s View

The dispatcher’s view is used to present the data (topology and current status) to the dispatcher. It is also used to accept orders from the dispatcher which will be transferred to the Central Unit.

2.4 Infrastructure Communication Units

Every piece of infrastructure has to communicate with the Central unit. These are points, signals and track occupancy detectors. They will regularly report their status to the central unit and ask for demanded changes of their state. (Currently the Central unit only allows to deal with incoming messages or messages to devices known at time of implementation.) Another solution would be to create a communication unit that would be able to transmit messages to the infrastructure.

3 Rules of the Railway

3.1 Signals

Definition 1 A signal is a device which uses colour lights or other means to give information to the train driver.

Definition 2 An aspect or signal aspect encodes the information given to the train driver, as for example One green light above a yellow light. A signal book describes the appearance and meaning of a signal aspect.

Definition 3 A signal information is the actual information given to the driver, for example proceed with 40 km/h. A signal aspect may include more than one piece of signal information.

Signals are used to give instructions to the train driver. One, but not the only, type of signals are light signals, which show a coloured light or a combination of more than one light. The meaning depends on the local signal book. But generally a red lights stands for ‘stop’, and a green light for ‘proceed’.

3.2 Stations and open lines

Railways are divided into stations and open lines. A station is usually under supervision of a single dispatcher. Figure 2 shows an exemplary design of a railway station. The station has two main tracks with traffic running on the right. It has two sidings and two platforms and serves as a junction for the branching line.

An open line between two stations is under common supervision of the two adjacent supervisors. Two conditions have to be met to send a train onto the line. Firstly a permission from the following dispatcher is needed to agree who may send a train onto the line next. Secondly, if the permission is given, the line must be clear of the preceding train. This may be the distance to the next station or until an intermediate signal.

On a dual track line each dispatcher has the permission to send trains onto the line for one of the tracks. The permission may be transferred if necessary on dual track lines, and are commonly transferred on single track lines. Lines consisting of more than
two tracks can be logically split into several lines with one or two tracks. The information about the line being free can be gathered manually from the following dispatcher or from track occupancy detectors. If the line is split into segments separated by automatic block signals the track is only required to be free until the next signal. As a permission may only be transferred if no train is along the line, it is assured that all trains drive into the same direction.

A station is characterized by entry and exit signals. A train may arrive at an entry signal at any time, so the dispatcher cannot control the arrival of trains, unless permission for the preceding station is revoked. A train passing an exit signal will proceed on an open line. At smaller stations, usually called junctions, one signal serves as entry and exit at the same time, so the trains proceed from one open line to another directly.

3.3 Definition of train movements

A train movement generally begins and ends at a signal. It may also end on an approach to the open line. When all conditions, which will be explained later, are met, the train is allowed to proceed from one signal to the next.

3.4 Basic Approach

Definition 4 The start signal of a train movement is located at the start of the movement. It is facing the train before the movement is started. It will change from a stop aspect to a proceed aspect, when the movement is allowed. It will change back to stop, when the train has passed the signal.

Definition 5 The target signal of a train movement is located at the end of the movement. If no subsequent route is set, the train has to stop at that signal.

Definition 6 A track element is any part of the tracks required to determine if the route can be set. These are amongst others points, signals, track occupancy detectors, isolators or logical elements like a distance marker.

Definition 7 A track segment is a part of the track, which is bounded by isolators and equipped with a track occupancy detector. The detector will be able to determine if there is a train anywhere on the tracks covered. Only this bit of information is required, even if the detector may be able to gather more information like speed or direction. A track segment consists of one or more track elements.

A train movement consists of a start and a target signal. It further includes a number of track segments which must be neither occupied or part of a conflicting movement. The track segments include a list of track elements which must be in a given condition. When a train movement is allowed, all track elements are locked to that movement. The train will proceed to the target signal, when the signal is open. When the arrival is detected, the train movement is completed and all track elements are unlocked.

A train movement contains information about the track segments to be passed. One or more of these are declared as target. When a train occupies no other track segments but these, after entering and leaving all previous track segments in the given order, it can now be assumed that the train has arrived. A clock is ticking now for a given time to ensure the train has stopped or is at a relatively low speed. When this extra time has passed the train is considered to have stopped. The movement is finished now and has no longer got any influence on other routes. No information about the train or the movement is retained. The only difference between the previous and the present state is that the occupied track segments have changed.
4 Modelling

4.1 Editor Model

This model has to contain all information needed to calculate the interlocking model, as well as some geographical and topological data which support the comprehension of the user.

The editor will have a grid, where the track elements can be placed onto a free grid position. Tracks can be put horizontally or diagonally. Many track elements are restricted to a horizontal alignment. Figure 3 shows different types of points. These symbols may be flipped horizontally or vertically but not rotated. Apart from simple straight and curved tracks there are many other elements like points, signals, isolators, overlap markers or track occupancy detectors. Auxiliary signal elements give additional information about restrictions to be shown at the main signal or intermediate signals. Figure 4 shows a possible layout for the station from Figure 2. For the editor model to be correct, every track element has to have all of its connectors in use. The editor model will now be translated into the interlocking model. This is going to be a topological model. The straight and curved tracks as well as any decoration element will therefore be ignored. Isolators and elements with built-in isolators, as well as track occupancy detectors, will be used to create groups. Movements will be created by starting a search form each signal.

![Figure 3: Track elements for points](image)

![Figure 5: Possible routes between two signals.](image)

Different possible overlaps further multiply the number of routes between the two signals. A priority has to be set to be able to present the most useful routes to the train dispatcher without prohibiting the routes of minor importance.

One possible scoring can be found in [2], which only selects the route with the least number of points passed. This model will keep all possible routes instead, while rating the routes for the number of points passed in reverse position, the maximum speed allowed and the number of conflicting routes. Equation 1 shows a simple approach to assign a priority to a route. Nevertheless, as there is no natural or objective order, any other priority assignment may yield a different order. Therefore it must be possible for a human to reorder the list. One possibility would be to monitor the dispatcher to adapt the order to his preferences.

\[
P = n_R w_R + v w_V - n_M w_M
\]

where
\[
\begin{align*}
    n_R & := \text{Number of points reversed} \\
    w_R & := \text{Weight for points reversed} \\
    v & := \text{Maximum speed allowed [10km/h]} \\
    w_V & := \text{Weight for maximum speed} \\
    n_M & := \text{Number of conflicting speed} \\
    w_M & := \text{Weight for conflicting movement} \\
    P & := \text{Priority}
\end{align*}
\]
4.2 Interlocking Model

Figure 6: Interlocking Model

The interlocking model as seen in Figure 6 consists of track elements $\tau$ in set $T$.

$$T = \{\tau_1, ..., \tau_n\}$$  \hspace{1cm} (2)

The track elements are grouped to track segments $\theta$ with $\Theta$ as the set of all track segments.

$$\Theta = \{\theta_1, ..., \theta_n\}$$  \hspace{1cm} (3)

$$\theta_i = \{\tau_{i1}, \tau_{i2}, \ldots\}$$  \hspace{1cm} (4)

A track element can be paired with a condition $c$. These can be grouped with a track segment to a condition set $\theta'_i$.

$$\theta'_i = \{\theta_j, \{\tau_{k1}, c_{k1}\}, \ldots, \{\tau_{kn}, c_{kn}\}\}$$  \hspace{1cm} (5)

$$\text{seg}(\theta'_i) := \theta_j$$  \hspace{1cm} (6)

$$\text{cnd}(\theta'_i) := \{\{\tau_{11}, c_{11}\}, \ldots, \{\tau_{n1}, c_{n1}\}\}$$  \hspace{1cm} (7)

$$\text{cnd}(\theta'_i, x) := \{\tau_x, c_x\}$$  \hspace{1cm} (8)

$$\text{size}(\theta'_i) := n$$  \hspace{1cm} (9)

These condition sets are collected in lists $\Theta'$.

$$\Theta'_i = \{\theta'_i1, ..., \theta'_in\}$$  \hspace{1cm} (10)

A train movement is defined by a start signal, a target signal and three sets of conditions with conditions. They are grouped into track segments, which don’t serve as a possible target, those which do and those serving as overlap. The two signals are from the set of track elements. All available train movements form the set $M$.

$$m = \{\tau_{\text{start}}, \tau_{\text{target}}, \Theta'_j\text{way}, \Theta'_j\text{target}, \Theta'_j\text{overlap}\}$$  \hspace{1cm} (11)

$$M = \{m_1, ..., m_n\}$$  \hspace{1cm} (12)

The sets $M$, $T$, and $\Theta$ and its elements are constant.

$$M, T, \Theta \text{ const.}$$  \hspace{1cm} (13)

An active train movement is characterised by the movement concerned and a list of track segments which are still to be occupied or to be left unoccupied after having been occupied. The active train movements are collected in set $M'$. Both $\Theta$ are initialised with all $\theta$ of $\Theta'_j\text{way} \cup \Theta'_j\text{target}$.

$$m' = \{m, \Theta_{\text{occupy}}, \Theta_{\text{unoccupy}}\}$$  \hspace{1cm} (14)

$$M' = \{m_1, ..., m_n\}$$  \hspace{1cm} (15)

Every track element is in one state $s$ out of a set of possible states $S$.

$$S = \{s_1, ..., s_n\}$$  \hspace{1cm} (16)

Every track element can be required to meet conditions $c$. The function $\text{cmp}$ checks if the state $s$ meets the conditions.

$$\text{cmp}(c, s) \in \{\text{true, false}\}$$  \hspace{1cm} (17)

The active condition sets of an active movement $m'$ are those form $\Theta'_j\text{way}$ and $\Theta'_j\text{target}$ whose track segment is in $\Theta_{\text{occupy}}$ or in $\Theta_{\text{unoccupy}}$, as well as those from $\Theta'_j\text{overlap}$.

$$C_{m'} = m'.m.\Theta'_{\text{overlap}} \cup \{\theta' \in (m'.m.\Theta'_{\text{way}} \cup m'.m.\Theta'_{\text{target}}) \text{seg}(\theta') \in (m'.\Theta_{\text{occupy}} \cup m'.\Theta_{\text{unoccupy}})\}$$  \hspace{1cm} (18)

Therefore, the list of all active condition sets is

$$C_{M'} = \bigcup_{m' \in M'} (C_{m'})$$  \hspace{1cm} (19)

A track element’s active conditions $C_{\tau_n}$ can be determined by finding those condition pairs from the condition sets in $C_{M'}$, whose track element is equal to $\tau_n$

$$C_{\tau_n} = \left\{ a.c \mid a \in \bigcup_{k \in C_{M'}} \text{cnd}(k) \land a.\tau = \tau_n \right\}$$  \hspace{1cm} (20)

From this set of conditions a new status for the element should be determined. Therefore a function $\text{get\ State}$ should be developed. The function returns the possible states which fulfil all conditions as a list, sorted by priority.

$$S_{\text{new}} = \text{get\ State}(C_{\tau_n})$$  \hspace{1cm} (21)
This function can be used to determine if a new condition \( c \) is not in conflict with a set of conditions \( C \).

\[
\text{conflict}(c, C) := \text{getState}(C \cup \{c\}) = \{\}
\]

(22)

The model would be faulty if any track segment has no possible state.

\[
\forall \tau \in T \ (\text{getState}(C_\tau) \neq \{\})
\]

(23)

The user action of setting a new route can be described as adding a new movement \( m'_F \) to \( M' \). This causes new condition sets to be added to \( C_{M'} \). This must not be in conflict with Equation 23. Therefore, a function is needed to detect a conflict before adding the movement. The set \( M'_F \) describes the possible future set in place of \( M' \)

\[
M'_F = M' \cup \{m'_F\}
\]

(24)

The set of active conditions would be joined by conditions from \( m'_F \)

\[
C_{M'_F} = \bigcup_{m' \in M'_F} (C_{m'}) = \bigcup_{m' \in M' \cup \{m'_F\}} (C_{m'})
\]

\[
= \left( \bigcup_{m' \in M'} (C_{m'}) \right) \cup C_{m'_F} = C_{M'} \cup C_{m'_F}
\]

(25)

The new conditions for a given track element would be analogous to Equation 20

\[
C_{F \tau_n} = \left\{ a.c \big| a \in \bigcup_{k \in C_{m'_F}} \text{cmd}(k) \land a.\tau = \tau_n \right\}
\]

(26)

A conflict of \( m'_F \) and \( M' \) would now be defined as

\[
\text{conflict}(m'_F, M') := \neg (\forall \tau \in T (\text{getState}(C_{F\tau}) \neq \{\}))
\]

\[
= \exists \tau \in T (\text{getState}(C_{F\tau}) = \{\})
\]

(27)

As the model was in a correct state before, only elements of \( m'_F \) may cause a conflict. So only those track elements must be checked. When no conflict is detected, the movement is added to \( M' \).

The software will regularly compare status \( s \) of an element with the list of possible new states \( S_{new} \). If the current state is not in the list of possible states, or another state form \( S_{new} \) has a higher priority, the software will try to change the state of the element. If that happens, the element will receive the attribute ‘busy’, which is maintained until the new state is in effect to prevent continuously sending the same command to the element. No attempt to change the state will be undertaken, if the track segment to which the element belongs to is occupied or reserved.

When all conditions for a movement are met, the route is active, which means the signal \( \tau_{start} \) is opened.

### 4.3 Controller

The User will have access to two basic functions of the controller. The first function is ‘Allow Train Movement’. This function requires an identification for the train movement. The identification selects a movement \( m \in M \). It will create a proposed movement \( m'_F \) and check for a conflict using Eq. 27. If no conflict is detected, \( m'_F \) will be added to \( M' \). This implies an initiation of changes of the state of the infrastructure, which in turn will have the signal opened. This function could return data, whereas it is not necessary as the information from the second function ‘Get Status’ will give evidence of a successfully set route. This function requests information about the status of logic and infrastructure. This information will be used by the GUI to display the status. It will be a read-only function.

### 4.4 Point model and behaviour

A point is in one of three states: normal, reversed or unknown. Two out of these three states (normal and reversed) can be requested. Furthermore, the point may be busy or it may be faulty. This additional information helps to determine whether the route is still trying to be set or if it finally failed. The request-able states are given a priority order. By default the priority of normal is higher than reversed, but that can be set for every point individually.

The conditions \( c \) for a point is a combination of two properties. This is the direction (normal or reversed) and a priority (mandatory or optional). That means that there are four different conditions. The set of all possible combinations of conditions would therefore have the size of sixteen. Some of them are shown in Table 1.

### 4.5 Behaviour

The most important feature of the CCU is to be able to allow, block and release a train movement. In order to allow the movement the track segments along the route have to be checked whether they are free to be passed. The points are checked as well even if that should be redundant. Additionally, the target signal
Table 1: States resulting from a given condition set

<table>
<thead>
<tr>
<th>Set of Conditions</th>
<th>List of States</th>
</tr>
</thead>
<tbody>
<tr>
<td>{ }</td>
<td>{n, r}</td>
</tr>
<tr>
<td>{nm}</td>
<td>{n}</td>
</tr>
<tr>
<td>{ro}</td>
<td>{r, n}</td>
</tr>
<tr>
<td>{no, rm}</td>
<td>{r}</td>
</tr>
<tr>
<td>{nm, rm}</td>
<td>{}</td>
</tr>
</tbody>
</table>

n = normal, r = reversed, m = mandatory, o = optional

will be checked. It might still be open from an earlier movement that has passed. The track segments belonging to the overlap must be free of conflicting movements. That means, a concurrent overlap or a consecutive movement is allowed. The same is true for the first segments after the start signal which may be used by an overlap from a preceding movement to the start signal.

When the movement is possible, it is immediately locked logically to avoid a second movement to be allowed while the first is believed to be free. This is done by adding the movement to the list $M'$. Whenever a condition is detected to be finally failed to be met, the movement is removed from the list $M'$. If the enabling works, the last step is to open the signal. The movement now can no longer be made undone.

When the train moves past the start signal the process of releasing the movement is started. The signal will be closed again, when the first segment is occupied. The track segment is removed from $\Theta'_{\text{occupy}}$ if it is occupied and it is the first element in that list. Once it is clear after being assumed to be occupied by the train the reservation on it is removed. The track segment is removed from $\Theta'_{\text{anoccupy}}$ if it is not occupied, the first element in that list and not in the list $\Theta'_{\text{occupy}}$. It is not explicitly know which movement is affected when a track segment is occupied or unoccupied. As any track segment is to be occupied by only one movement at a time the algorithm will find the movement concerned by searching all $\Theta'$ in $M'$. Once there are no more non-target segments left reserved but unoccupied, which means that both $\Theta'$ only contain segments marked as possible targets, a timer will start. If any change in $\Theta'$ occurs, the timer will be reset.

When the timer runs out, the movement is considered finished and will be deleted.

4.6 Signal aspects

A signal aspect can be separated into a main signal part showing information about the block following the next signal. If no route is set form a signal it will show stop. If a route is set from a signal to the next, the main part of the aspect is independent from the state of the next signal. Independent from the visual appearance, a separate information for main and distant signal must be maintained. The information on a German signal can be represented as a combination of allowed speed in steps of 10 km/h including zero representing stop, an optional direction letter and an optional left-track-indicator. Depending on the type (main, distant or both) of the signal and the signal system, an aspect is determined by this information. The information for the main signal part for each movement will be calculated by the editor model and will be included into the signal box model as an attribute of a movement $m$.

5 Example

The following example will show how the described algorithms explained above affect the station from Figure 2. The track segments are labelled with a leading $G$. Points have a leading $W$. The remaining labels $A, F, G, X, Y, Z$ and those with a leading $P$ or $N$ are main signals. The track map can be transformed into a graph. Every piece of information from the track map will be transformed into nodes. These are connected by edges, which contain no further information.

Figure 7: Track Map

The nodes are divided into three types, which are used to group the edges. All edges which connect to a node with a black/white dot via black nodes only
form a group, which is named after that node. Isolator nodes are shown with a gray node.

<table>
<thead>
<tr>
<th>Track segment</th>
<th>Element</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>G 52</td>
<td>G 52</td>
<td>unoccupied, unreserved</td>
</tr>
<tr>
<td>G 32</td>
<td>G 32</td>
<td>unoccupied, unreserved</td>
</tr>
<tr>
<td></td>
<td>Point W 1</td>
<td>normal, locked</td>
</tr>
<tr>
<td></td>
<td>Point W 2</td>
<td>normal, locked</td>
</tr>
<tr>
<td></td>
<td>Point W 3</td>
<td>normal, locked</td>
</tr>
<tr>
<td>G 2</td>
<td>G 2</td>
<td>unoccupied, unreserved</td>
</tr>
<tr>
<td></td>
<td>Signal N1</td>
<td>working correctly</td>
</tr>
<tr>
<td></td>
<td>Overlap</td>
<td>G 22 unoccupied, unreserved except concurrent overlaps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>G 42 unoccupied, unreserved except concurrent overlaps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Point W 10 normal, optionally locked (=normal or unlocked)</td>
</tr>
</tbody>
</table>

The conditions shown in Table 2 can be deducted from the topology map. The path of the train from A to N1 with a standard overlap assuming a maximum speed of 160 km/h will lead from the node A to the node N1 with the overlap continuing to node End of overlap N1. The path until N1 is split into track segments to allow the track elements to be released as soon as possible. The segments involved are G 52, G 32 and G 2. The overlap includes segments G 42 and G 62, which don’t form a group of their own but are in group “Overlap”. G 32, for instance, also includes three points that all have to be in a normal and locked position. Point W1 is not part of the group by being in the G 32 topology group, but as a side protection requirement from point W2, which indeed is in that group.

6 Conclusion

In this paper a set of railway rules were defined. According to that, a model of a blank interlocking software was then created to fit the railway rules, whereas the software is intended to fulfill the requirements of functional safety. A station layout that should serve as an example was investigated and used to create a part of the signal box model. Solutions were found for a way to deal with the concept of a partial release of routes, and it has been figured out how information on the signal aspect shown at the entrance of each route can be gathered.

7 Perspective

Some aspects of this approach were isolated and tested. The next step will be the creation of a software implementing the full model specified in section 4. A description and extension of the elements of the editor model will be given in a following paper. An algorithm has to be found to create the signal box model out of the editor model. Also, the design of the dispatcher’s view should be generated from the editor model.

References:


