International Journal of Innovative Computing, Information and Control
Volume **8**, Number **8**, August **2012**

ICIC International ©2012 ISSN 1349-4198
pp. 5677–5690

TOWARDS THE SAFETY PROPERTIES OF MOVING BLOCK RAILWAY INTERLOCKING SYSTEM

Nazir Ahmad Zafar[1], Sher Afzal Khan[2] and Keijiro Araki[3]

[1]Department of Computer Science
King Faisal University
PO Box 400, Hofuf, Saudi Arabia
nazafar@kfu.edu.sa

[2]Department of Computer Sciences
COMSATS Institute of Information Technology
Attock, Pakistan
sherafzal@ciit-attock.edu.pk

[3]Graduate School of Information Science and Electrical Engineering
Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan
araki@csce.kyushu-u.ac.jp

Received April 2011; revised August 2011

Abstract. *Railway interlocking is a safety critical system because its incorrect functioning may cause serious consequences. Modeling of a reliable interlocking has become a challenging problem due to its inherent complexity and introduction of new technologies. In this paper, formal analysis of safety properties of moving block interlocking is presented preventing collision and derailing of trains at the critical components of the network. We have supposed that the existence of two trains at a component is a collision. If the train's direction and switch control are inconsistent then it is assumed derailing at the switch. A step-by-step procedure is proposed to analyze the safety properties reducing complexity of the system using graph theory and Z notation. Initially, we defined the abstract safety properties, and then they are redefined by introducing a notion of moving block. Further, the safety properties are analyzed and extended by introduction of computer based controls. The formal specification is analyzed and validated using Z/Eves tool.*

Keywords: Railway interlocking system, Moving block control, Safety properties, Formal methods, Z-specification

1. Introduction.

A safety critical system is one whose incorrect functioning may have very serious consequences. As the use of computers in safety critical systems such as aircraft controls, medical instruments, defense systems, nuclear power plants, has been increased, concern for the safety of those systems has also grown up. Much of this concern has been focused on the software component of those computer-based systems. The use of formal methods is recommended to achieve required confidence level in the development of such systems because of having an exhaustive computer tool support and use in many applications as in [9, 27]. As railway interlocking is a safety critical system, hence it requires advanced formal modeling techniques and step-by-step development to ensure safety, quality and reliability of the system.

The huge train's weight in tons, great length in thousands of meters, iron track, distributed environment and presence of critical components make the moving block interlocking a highly complex system for its modeling, verification and simulation. There exists much work on modeling of the interlocking system. The work carried in [1, 5, 10, 14, 21] and a list [4] of 299 publications, addressing various issues on this topic, proves its importance. Most of the publications are on the traditional fixed block, whereas this work on moving block interlocking reflects a different approach. Hansen [11] uses Vienna development method (VDM) to model concepts of railway topology. Safety criteria are defined and validated through simulation but again his work is for fixed block interlocking. In [24], the system is constructed in stages by making both control and information data coexist on the same network for modeling a large-scale system. X. Hei et al. [33] describe a railway safety system that focuses on electronic interlocking system technology. Chevillat [6] proposes a model based generation of interlocking controller software from control tables using a model-driven tool chain. She et al. [28] present two styles of interlocking using graph search approaches based on revised Dijkstra and heuristic algorithms. In [5], an approach to testing interlocking is proposed and simulated. Khan et al. specify the moving block using a strategy of promotion using Z in [18]. Harrison et al. [12] discuss the preparation of requirements specification for a railway signalling interlocking using Z notation. Ales Janota [16] discuss the use of Z notation in requirement specification of the railway interlocking system behavior. In [15], a workflow of analyzing the dynamic behavior of safety critical embedded systems using HySAT is explored. The trains scheduling strategy of concurrent request in a distributed railway interlocking system is discussed in [13]. Some other relevant work can be found in [2, 3, 7, 8, 19, 25, 26, 29, 32].

Major objective of this research is developing a cost-effective, safe, efficient and verified computer model for the moving block railway interlocking. Some preliminary results of our research were presented in [34, 35, 36] where few errors and inconsistencies are identified at some places. The work presented in [34] describes a formal specification of the static aspect of the system. Further, we presented a detailed analysis of the critical components by using integration of graph theory and Z notation in [35]. In this research, formal definitions of static system addressed in the previous work are reused after some improvements to describe the formal analysis of the safety of the system. The most important part of our contribution in this work is formal description of the safety properties preventing collision and derailing of trains. We present four safety properties, three for avoiding collision and one for derailing of trains. We analyze and formalize the properties at three different levels, i.e., from abstraction to detailed analysis. Initially, we define the abstract safety properties at linear track, crossing, level crossing and switch. Then we redefine the abstract properties by associating a concept of moving block with a train. Finally, we introduce a computer based control system on the redefined properties for the further formal analysis and description. Formal specification of the system is analyzed and validated using Z/Eves tool. Rest of the paper is organized as follows.

In Section 2, an introduction to railway interlocking system and formal methods is presented. In Section 3, we define and formalize the abstract safety properties. In Section 4, formal specification of the redefined properties is described. In Section 5, we use computer based control systems to further analyze and formalize the safety properties. Finally, conclusion and future work are given in Section 6.

2. Railway Interlocking System and Formal Methods.

The purpose of railway interlocking system is to prevent trains from collisions and derailing while at the same time allowing normal train's movement. There are various types of interlocking systems, in practice, varying from purely mechanically operated to state-of-the-art computerized moving block interlocking systems. Fixed block and moving block interlocking are two main types of interlocking. Moving block is gaining more important in railway industry due to some disadvantages in fixed block interlocking system.

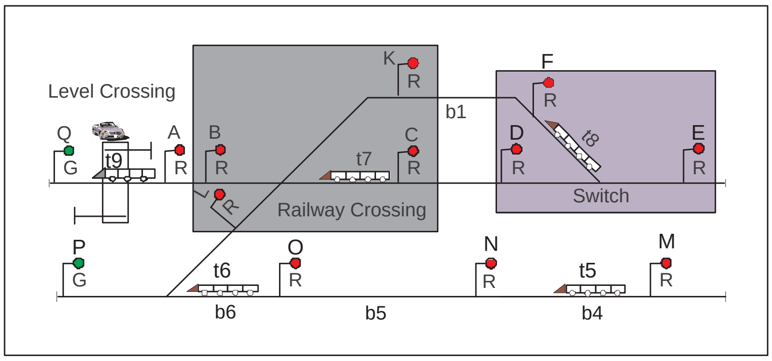

Figure 1. Fixed block railway interlocking system

The fixed block interlocking system divides the railway network into fixed blocks and components which are separated by signals but are highly dependant on each other as shown in Figure 1. To understand the dependance, consider the same figure in which the train t5 can enter block b5 only when the train t6 has cleared the block b6; otherwise there may be a collision. This means that there is always a distance of more than one block between two trains at linear tracks. The railway crossing consists of two crossovers, if a train occupies one the other crossover must be protected. Further, the railway crossing is dependant upon the level crossing next to it. As a result, the biggest drawback of the system is that a long distance is required between two trains which limit the capacity of railway line and speed of a train making it a highly inefficient system. In reality, the safe distance between two trains is the distance needed for a train to come to a complete stop, which is much less than the length of a fixed block and even shorter if a train is moving at a low speed. Whereas in the moving block technology, instead of cutting a piece of railway line into fixed blocks, the train's occupying area and some distance in front of it becomes the moving block in which no other train can enter. The area in front of a train is at least its braking distance. In this way, trains can move much closer to each other making it possible to operate at higher speeds and frequencies. As the system under hand is a critical and complex one, hence, it requires much advanced techniques for modeling and then proving its correctness and completeness. Formal methods are applied in this research, in terms of Z notation, because their use is recommended to achieve required confidence level in safety critical systems. The Z is a model oriented approach based on the standard mathematical notations used for the specification of abstract properties unlike a detailed description language [31]. The Z/Eves is one of the powerful tools used for the analysis of Z specification which will be used to analyze the system's schema expansion, pre-condition calculation, domain checking, syntax and type checking, and theorem proving.

3. **Formalizing Abstract Safety Properties.** Collision and derailing are defined as: existence of two trains on a track segment is a collision; existence of two trains on a crossing is a collision; when a train is at level crossing and barriers are open, is a collision; if direction of a train and state of a switch do not match, it may cause derailing. Based on these definitions, the safety properties can be stated as: (i)-(ii) There must be, at most, one train at one track segment and crossing to avoid collision. (iii) Barriers must be closed when a train is at level crossing to avoid collision between train and road traffic. (iv) A train must respect state of a switch control preventing derailing.

3.1. **Formal static model.** Few constructs of formal model described in [35] are presented below to be used for the analysis of safety properties. It is to be noted that *root* defined below in the schema *Switch* is an edge, and *crossover1* and *crossover2* defined in schema *Crossing* are set of edges. In the schema *TrainInfo* a variable *total* is added to know the total number of trains at a track segment. In *Control* schema, variable *capacity* is included to define the maximum number of trains which are possible in a section. Further, speed and destination variables are removed from *Train* schema because it does not require at this level of specification. It is mentioned that *StaticModel* and *DynamicModel* schemas are used without *Xi* notation.

3.2. **Formal specification of abstract safety properties.** The specification of the safe system denoted by *IsSafe* is composed of two schemas, i.e., *NoCollision* and *NoDerailing*.

$$[\mathit{Node}];\ \mathit{Graph} == \{x : \mathit{Node};\ y : \mathit{Node} \mid x \neq y \bullet (x, y)\}$$
$$\mathit{SwitchControl} ::= \mathit{LEFT} \mid \mathit{RIGHT}$$

$$\begin{array}{|l}\hline \mathit{Switch} \\ \mathit{switch} : \mathit{Node} \\ \mathit{root} : \mathit{Graph} \\ \mathit{control} : \mathit{Node} \times \mathit{Node} \\ \qquad \to \mathit{SwitchControl} \\ \hline \end{array} \qquad \begin{array}{|l}\hline \mathit{Crossing} \\ \mathit{cross} : \mathit{Node} \times \mathit{Node} \\ \mathit{crossover}1, \mathit{crossover}2 : \mathbb{P}\,\mathit{Graph} \\ \hline \end{array}$$

$$\begin{array}{|l}\hline \mathit{LevelCrossing} \\ \mathit{levelCrossing} : \mathit{Node} \\ \hline \end{array}$$

$$\mathit{Switches} == \mathit{seqSwitch};\ \mathit{Crossings} == \mathit{seqCrossing}$$
$$\mathit{LevelCrossings} == \mathit{seqLevelCrossing};\ [\mathit{TrainId}]$$
$$\mathit{TrainType} ::= \mathit{FIXEDROUTE} \mid \mathit{AUTONOMOUS}$$
$$\mathit{TrackStates} == \mathit{Node} \nrightarrow \mathit{TrainInfo}$$
$$\mathit{State} ::= \mathit{CLEAR} \mid \mathit{OCCUPIED};\ \mathit{Operation} ::= \mathit{INTERLOCK} \mid \mathit{MANUAL}$$

$$\begin{array}{|l}\hline \mathit{StaticModel} \\ \mathit{topology} : \mathbb{P}\,\mathit{Graph};\ \mathit{switches} : \mathit{Switches} \\ \mathit{crossings} : \mathit{Crossings};\ \mathit{levelcrossings} : \mathit{LevelCrossings} \\ \hline \end{array}$$

$$\begin{array}{|l}\hline \mathit{TrainInfo} \\ \mathit{trainId} : \mathit{TrainId} \\ \mathit{trainType} : \mathit{TrainType} \\ \mathit{total} : N \\ \hline \end{array} \qquad \begin{array}{|l}\hline \mathit{SwitchState} \\ \mathit{switchState} : \mathit{State} \\ \mathit{occupiedBy} : \mathit{TrainId} \\ \mathit{swControl} : \mathit{SwitchControl} \\ \mathit{operation} : \mathit{Operation} \\ \hline \end{array}$$

$$\mathit{SwitchStates} == \mathit{Node} \to \mathit{SwitchState}$$
$$\mathit{CrossStates} == \mathit{Node} \times \mathit{Node} \to \mathit{CrossState}$$

$$\begin{array}{|l}\hline \mathit{CrossState} \\ \mathit{crossState} : \mathit{State} \\ \mathit{occupiedBy} : \mathit{TrainId} \\ \hline \end{array} \qquad \begin{array}{|l}\hline \mathit{LevelState} \\ \mathit{levelState} : \mathit{State};\ \mathit{occupiedBy} : \mathit{TrainId} \\ \mathit{barriers} : \mathit{Barriers} \\ \hline \end{array}$$

$$\mathit{LevelStates} == \mathit{Node} \to \mathit{LevelState}\ \mathit{Barriers} ::= \mathit{CLOSED} \mid \mathit{OPEN};\ \mathit{MovingBlock} == \mathbb{P}\,\mathit{Graph}$$

$$\begin{array}{|l}\hline \mathit{DynamicModel} \\ \mathit{trackStates} : \mathit{TrackStates} \\ \mathit{switchStates} : \mathit{SwitchStates} \\ \mathit{crossStates} : \mathit{CrossStates} \\ \mathit{levelStates} : \mathit{LevelStates} \\ \hline \end{array} \qquad \begin{array}{|l}\hline \mathit{Train} \\ \mathit{movingBlock} : \mathit{MovingBlock} \\ \mathit{destination} : \mathit{Node} \\ \mathit{currentSpeed} : N \\ \mathit{speedLimit} : N \\ \hline \end{array}$$

$$\mathit{Trains} == \mathit{TrainId} \to \mathit{Train};\ \mathit{Section} == \mathbb{P}\,\mathit{Node};\ \mathit{States} == \mathit{Node} \to \mathit{State}$$

$$\begin{array}{|l}\hline \mathit{Control} \\ \mathit{section} : \mathit{Section} \\ \mathit{states} : \mathit{Node} \to \mathit{State} \\ \mathit{trains} : \mathit{Trains};\ \mathit{capacity} : N \\ \hline \end{array} \qquad \begin{array}{|l}\hline \mathit{RISSystem} \\ \mathit{StaticModel} \\ \mathit{DynamicModel} \\ \mathit{trains} : \mathit{Trains};\ \mathit{controls} : \mathit{Controls} \\ \hline \end{array}$$

$$[\mathit{ControlId}],\ \mathit{Controls} == \mathit{ControlId} \to \mathit{Control}$$

The schema *NoCollision* is used to describe safety properties for preventing collision at a network and the schema *NoDerailing* is used to define the property preventing derailing at a switch.
$IsSafe \mathrel{\widehat{=}} NoCollision \land NoDerailing$
where $NoCollision \mathrel{\widehat{=}} OneTrain1Track \land OneTrain1Xng \land LevBarWell$

3.2.1. *Safety at linear track.* The specification of *OneTrain*1*Track* assures that a track segment can be occupied by only one train at a time. The *TrackStates* is a mapping from *Node* to *TrainInfo*, where *TrainInfo* is a schema consisting of three variables that is train identifier, type and total number of trains at a track segment. In the schema, it is stated that for all track segments, the total number of trains at the segment should not be greater than one.

$$
\begin{array}{|l}
\textit{OneTrain1Track} \\
\hline
\Xi RISSystem \\
\hline
\forall t : TrainInfo \mid t \in \operatorname{ran} trackStates \bullet t.total \leq 1 \\
\hline
\end{array}
$$

3.2.2. *Safety at crossing.* Railway crossing is an intersection of two track segments. If there are two trains on crossing, one train at each segment, then the first property is satisfied but there is a possibility of collision. Collision at a crossing can be avoided if only one track segment of the crossing can be occupied at a time. In the schema *OneTrain*1*Xng*, it is specified that sum of the number of trains moving at the segments of the crossing should not be greater than one.

$$
\begin{array}{|l}
\textit{OneTrain1Xng} \\
\hline
\Xi RISSystem \\
\hline
\forall cr : Crossing;\ t1, t2 : TrainInfo \mid (cr.cross.1, t1) \in trackStates \\
\land (cr.cross.2, t2) \in trackStates \bullet t1.total + t2.total \leq 1 \\
\hline
\end{array}
$$

3.2.3. *Safety at level crossing.* The schema *LevBarWell* states that if a train occupies level crossing then its barriers must be closed for preventing collision.

$$
\begin{array}{|l}
\textit{LevBarWell} \\
\hline
\Xi RISSystem \\
\hline
\forall lxng : LevelCrossing;\ t : TrainInfo;\ ls : LevelState \\
\mid lxng \in \operatorname{ran} levelcrossings \land (lxng.levelCrossing, ls) \in levelStates \\
\land (lxng.levelCrossing, t) \in trackStates \\
\bullet (t.total = 1 \Rightarrow ls.barriers = CLOSED) \\
\hline
\end{array}
$$

3.2.4. *Safety at switch.* In the schema *NoDerailing*, it is described that if any of the two segments is occupied and is a branch of a switch then control of the switch must be in the direction of that segment. For example, if train t5 enters into track segment 5, and moves along the right of the switch then the segments 4 and 7 are set in occupied state as shown in Figure 2.

4. **Redefining the Safety Properties.** Existence of one train at one segment cannot guarantee to avoid collision. This is because if the segment is small and train is moving at high speed then it may not be possible to stop it at the same segment and it may cause collision. On the other hand, the abstract properties were necessary as a foundation to develop the safety properties to be applied to a real interlocking.

NoDerailing

$\Xi RISSystem$

$\forall\, sw : Switch;\; ss : SwitchState;\; s : Node$
$\mid sw \in \operatorname{ran} switches \wedge (s, ss) \in switchStates$
$\bullet \forall\, sc : SwitchControl;\; s1, s2 : Node;\; t1, t2 : TrainInfo$
$\mid sc \in \operatorname{ran} sw.control \wedge (s1, s2) \in \operatorname{dom} sw.control$
$\wedge\, t1 \in \operatorname{ran} trackStates \wedge (s1, t1) \in trackStates \wedge (s2, t2) \in trackStates$
$\bullet\, t1.total = 1 \vee t2.total = 1 \Rightarrow sc = ss.swControl$

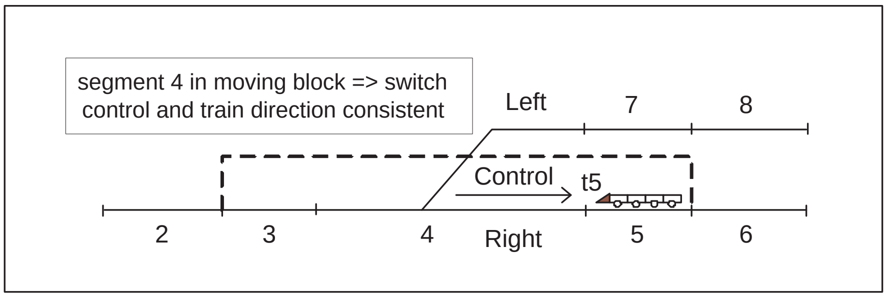
segment 4 in moving block => switch control and train direction consistent
Left
7
8
Control
t5
2
3
4
Right
5
6

FIGURE 2. Switch control

4.1. **Moving block and safety properties.** The safety properties are restated as (i) The intersection of moving blocks of two different trains is always empty. (ii) A crossing may be contained in the moving block of only one train. (iii) If a level crossing is in the moving block of a train then its barriers must be closed. (iv) If a switch is in the moving block of a train then its control must be consistent with the direction of the train.

4.2. **Formal specification of the safety properties.** The redefined safe system is denoted by the schema *IsSafeR* composed of two further schemas *NoCollisionR* and *NoDerailingR* as defined below.
$IsSafeR \mathrel{\widehat=} NoCollisionR \wedge NoDerailingR$; $NoCollisionR \mathrel{\widehat=} OneTrain1TrackR \wedge OneTrain1XngR \wedge LevBarWellR$

In the schema *OneTrain1TrackR*, it is defined that intersection of moving blocks of any two trains is always empty. In the formal specification, it is stated that for any element (s1, s2) in the moving block of train t1 and for any element (s3, s4) in the moving block of another train t2, intersection of the sets {s1,s2} and {s3,s4} must be empty.

OneTrain1TrackR

$\Xi RISSystem$

$\forall\, t1, t2 : TrainId;\; train1, train2 : Train$
$\mid (t1, train1) \in trains \wedge (t2, train2) \in trains$
$\bullet \neg t1 = t2 \Rightarrow train1.movingBlock \cap train2.movingBlock = \{\}$

In the schema *OneTrain1XngR*, it is specified that if a track segment of a railway crossing is in the moving block of one train then it cannot be contained in the moving block of another train. The horizontal schema *LevBarWellR* is further split into two schemas *TrainLXBarClosed* and *NoTrainLXBarOpen*. The first one schema *TrainLXBarClosed* is used to check safety at a level crossing if it is occupied by any train. The second schema *NoTrainLXBarOpen* is used to check if a level crossing is not occupied by any train then its barriers should not be closed unnecessarily.
$LevBarWellR \mathrel{\widehat=} TrainLXBarClosed \wedge NoTrainLXBarOpen$

OneTrain1XngR

$\Xi RISSystem$

$\forall t1, t2 : TrainId;\ train1, train2 : Train;\ xng : Crossing$
$\mid (t1, train1) \in trains \wedge (t2, train2) \in trains \wedge xng \in \operatorname{ran} crossings$
$\bullet \neg t1 = t2$
$\Rightarrow \neg(xng.crossover1 \cup xng.crossover2) \cap train1.movingBlock = \{\}$
$\Rightarrow (xng.crossover1 \cup xng.crossover2) \cap train2.movingBlock = \{\}$
$\wedge \neg(xng.crossover1 \cup xng.crossover2) \cap train2.movingBlock = \{\}$
$\Rightarrow (xng.crossover1 \cup xng.crossover2) \cap train1.movingBlock = \{\}$

TrainLXBarClosed

$RISSystem$

$\forall lxng : LevelCrossing;\ ls : LevelState$
$\mid lxng \in \operatorname{ran} levelcrossings \wedge (lxng.levelCrossing, ls) \in levelStates$
$\bullet \exists\, tid : TrainId;\ train : Train \mid (tid, train) \in trains$
$\bullet \forall s1, s2 : Node \mid (s1, s2) \in train.movingBlock$
$\bullet s1 = lxng.levelCrossing \vee s2 = lxng.levelCrossing$
$\Rightarrow ls.barriers = CLOSED$

NoTrainLXBarOpen

$RISSystem$

$\forall lxng : LevelCrossing;\ ls : LevelState$
$\mid lxng \in \operatorname{ran} levelcrossings \wedge (lxng.levelCrossing, ls) \in levelStates$
$\bullet \forall tid : TrainId;\ train : Train \mid (tid, train) \in trains$
$\bullet \forall s1, s2 : Node \mid (s1, s2) \in train.movingBlock$
$\bullet \neg s1 = lxng.levelCrossing \wedge \neg s2 = lxng.levelCrossing$
$\Rightarrow ls.barriers = OPEN$

In the schema *NoDerailingR*, it is specified that if any branch of a switch is in the moving block of a train then control of the switch must be left in the direction of movement of the train.

NoDerailingR

$RISSystem$

$\forall sw : Switch;\ s : Node;\ ss : SwitchState$
$\mid sw \in ranswitches \wedge (s, ss) \in switchStates$
$\bullet \forall s1, s2 : Node;\ sc : SwitchControl \mid ((s1, s2), sc) \in sw.control$
$\bullet \exists\, tid : TrainId;\ train : Train \mid (tid, train) \in trains$
$\bullet (s1, s2) \in train.movingBlock \vee (s2, s1) \in train.movingBlock$
$\Rightarrow sc = ss.swControl$

5. **Further Analysis of the Safety Properties.**

Finally, computer based control systems are introduced for observing trains and network state space for completing the formalization of safety properties. The definitions of collision and derailing are the same as those given in the previous section but with a difference that, here, trains are under computer based controls. This analysis is necessary because trains itself cannot operate safely, it requires some system to observe the trains and network state space. In the further formal analysis, consistency of network state is also checked. By consistency we mean that state of a track segment must be consistent in the entire system. A concept of locking a component is also introduced.

5.1. **Further analysis.** The schema *IsSafeFR* is a redefined form of the schema *IsSafeR* and is defined as $\mathit{IsSafeFR} \mathrel{\widehat{=}} \mathit{NoCollisionFR} \wedge \mathit{NoDerailFR}$. The schema *NoCollisionFR* is composed of *OneTrain1TrackFR*, *OneTrain1XngFR* and *LBStateWellFR* defining no collision properties under a control.
$\mathit{NoCollisionFR} \mathrel{\widehat{=}} \mathit{OneTrain1TrackFR} \wedge \mathit{OneTrain1XngFR} \wedge \mathit{LBStateWellFR}$

5.1.1. *Formal analysis of Property 1.* For further analysis, preventing collision along a linear track, the schema *OneTrain1TrackFR* is introduced which is divided into two other schemas *MovingBlockEmpty* and *StateConsistent*. The first schema *MovingBlockEmpty* specifies that for any two different trains under a control, no train can enter into the moving block of another train.
$\mathit{OneTrain1TrackFR} \mathrel{\widehat{=}} \mathit{MovingBlockEmpty} \wedge \mathit{StateConsistent}$
$\mathit{StateConsistent} \mathrel{\widehat{=}} \mathit{MBlockConsistent} \wedge \mathit{TracksConsistent}$

MovingBlockEmpty
$\Xi \mathit{RISSystem}$

$\forall\, cid : ControlId;\ control : Control \mid (cid, control) \in controls$
$\bullet \forall\, tid1, tid2 : TrainId;\ train1, train2 : Train$
$\mid (tid1, train1) \in control.trains$
$\wedge\ (tid2, train2) \in control.trains \bullet \neg tid1 = tid2$
$\Rightarrow (\forall\, s1, s2, s3, s4 : Node \mid (s1, s2) \in train1.movingBlock$
$\wedge\ (s3, s4) \in train2.movingBlock \wedge \{s1, s2\} \cap \{s3, s4\} = \{\})$

In the schema *MBlockConsistent*, it is specified that if a track segment is in the moving block of a train then its state must be occupied in both the mappings *trackStates* (describing state of entire system) and *states* (describing state of piece of network under control) as described below. A track segment in the mapping *states* is occupied if and

MBlockConsistent
$\Xi \mathit{RISSystem}$

$\forall\, cid : ControlId;\ control : Control \mid (cid, control) \in controls$
$\bullet \forall\, tid : TrainId;\ train : Train \mid (tid, train) \in control.trains$
$\bullet \forall\, s1, s2 : Node \mid (s1, s2) \in train.movingBlock$
$\bullet \exists\, cs : State \mid cs \in \operatorname{ran} control.states \wedge cs = OCCUPIED$
$\bullet (s1, cs) \in control.states \wedge (s2, cs) \in control.states$

only if the track segment is in the moving block of a train, or is a part of switch or crossing which is occupied by a train. In the schema *TracksConsistent*, it is described that if a track segment is occupied in the *trackStates* mapping then it must be occupied in the *states* mapping. On the other hand, if a track segment is clear in the *states* mapping then it must be clear in the *trackStates* mapping.

TracksConsistent
$\Xi \mathit{RISSystem}$

$\forall\, cid : ControlId;\ control : Control \mid (cid, control) \in controls$
$\bullet \forall\, s : Node \mid s \in control.section \bullet \exists\, ts : TrainInfo;\ cs : State$
$\mid ts \in \operatorname{ran} trackStates \wedge cs \in \operatorname{ran} control.states$
$\bullet (s, ts) \in trackStates \wedge (s, cs) \in control.states$
$\wedge\ (ts.total = 0 \Rightarrow cs = OCCUPIED) \wedge (ts.total = 0 \Rightarrow cs = CLEAR)$

$$\begin{array}{|l}
\textit{OneTrain1XngFR} \\
\Xi \textit{RISSystem} \\
\hline
\forall \textit{cid} : \textit{ControlId};\ \textit{control} : \textit{Control};\ \textit{cs} : \textit{State} \\
\mid (\textit{cid}, \textit{control}) \in \textit{controls} \land \textit{cs} \in \operatorname{ran} \textit{control.states} \\
\bullet \forall t1, t2 : \textit{TrainId};\ \textit{train}1, \textit{train}2 : \textit{Train} \\
\mid (t1, \textit{train}1) \in \textit{control.trains} \land (t2, \textit{train}2) \in \textit{control.trains} \\
\bullet \forall \textit{xng} : \textit{Crossing};\ \textit{cr}1, \textit{cr}2 : \textit{Node};\ \textit{xs} : \textit{CrossState} \\
\mid \textit{xng} \in \operatorname{ran} \textit{crossings} \land (\textit{cr}1, \textit{cr}2) = \textit{xng.cross} \\
\land ((\textit{cr}1, \textit{cr}2), \textit{xs}) \in \textit{crossStates} \\
\bullet \neg t1 = t2 \land \textit{cr}1 \in \textit{control.section} \land \textit{cr}2 \in \textit{control.section} \\
\Rightarrow (\neg(\textit{xng.crossover}1 \cup \textit{xng.crossover}2) \\
\cap \textit{train}1.\textit{movingBlock} = \{\} \Rightarrow (\textit{xng.crossover}1 \cup \textit{xng.crossover}2) \\
\cap \textit{train}2.\textit{movingBlock} = \{\}) \land \textit{xs.crossState} = \textit{OCCUPIED} \\
\land \textit{xs.occupiedBy} = t1 \land (\neg(\textit{xng.crossover}1 \cup \textit{xng.crossover}2) \\
\cap \textit{train}2.\textit{movingBlock} = \{\} \Rightarrow (\textit{xng.crossover}1 \cup \textit{xng.crossover}2) \\
\cap \textit{train}1.\textit{movingBlock} = \{\}) \land \textit{xs.crossState} = \textit{OCCUPIED} \\
\land \textit{xs.occupiedBy} = t2 \\
\hline
\end{array}$$

5.1.2. *Formal analysis of Property 2.* In the schema *OneTrain1XngFR* given above, it is described that for any two trains under a computer based control system, a crossing is picked and its critical area is deduced. Then it is checked if the crossing is in the section under the control. If the crossing is in the moving block of one train, then it cannot be occupied by any other train and it must be locked. The property is described for all the trains and crossings.

5.1.3. *Formal analysis of Property 3.* The third property is described using the schema *LBStateWellFR* which depends on two further schemas *LBStateWell* and *MBConsistentAtLX*. In schema *LBStateWell*, relationship between level crossing and its barriers is defined. In schema *MBConsistentAtLX*, it is checked that if level crossing is in the moving block of a train then it must be locked. The schema *LBStateWell* is composed of two schemas *TrainAtLXBarClosedR* and *NoTrainAtLXBarOpenR* (not safety property). The schema *TrainAtLXBarClosedR* is used to check (i) if a level crossing is in the moving block of a train under given control, (ii) the level crossing is in the section under the same control. If both conditions are satisfied then the barriers must be closed.

$$\begin{array}{|l}
\textit{TrainAtLXBarClosedR} \\
\Xi \textit{RISSystem} \\
\hline
\forall \textit{cid} : \textit{ControlId};\ \textit{control} : \textit{Control};\ \textit{lxng} : \textit{LevelCrossing} \\
\mid (\textit{cid}, \textit{control}) \in \textit{controls} \land \textit{lxng} \in \operatorname{ran} \textit{levelcrossings} \\
\bullet \exists \textit{tid} : \textit{TrainId};\ \textit{train} : \textit{Train};\ \textit{ls} : \textit{LevelState} \mid (\textit{tid}, \textit{train}) \in \textit{control.trains} \\
\land (\textit{lxng.levelCrossing}, \textit{ls}) \in \textit{levelStates} \\
\bullet \exists s1, s2 : \textit{Node} \mid (s1, s2) \in \textit{train.movingBlock} \\
\bullet (s1 = \textit{lxng.levelCrossing} \lor s2 = \textit{lxng.levelCrossing}) \\
\land \textit{lxng.levelCrossing} \in \textit{control.section} \Rightarrow \textit{ls.barriers} = \textit{CLOSED} \\
\hline
\end{array}$$

$\textit{LBStateWellFR} \mathrel{\widehat=} \textit{LBStateWell} \land \textit{MBConsistentAtLX}$
$\textit{LBStateWell} \mathrel{\widehat=} \textit{TrainAtLXBarClosedR} \land \textit{NoTrainAtLXBarOpenR}$

In the schema *MBConsistentAtLX*, it is checked if a level crossing is locked, occupied, occupied by the right train and the barriers are closed. It is also verified that the segment identifying level crossing must be in occupied state.

5.1.4. *Formal analysis of Property 4.* The function *NoDerailFR* is used to analyze the fourth property, which is composed of three functions *NoDerailCFR*, *NoDerailAtAll* and

$$\begin{array}{l}
\textit{MBConsistentAtLX} \\
\hline
\textit{RISSystem} \\
\hline
\forall\, cid : ControlId;\ control : Control;\ cs : State;\ lxng : LevelCrossing \\
\mid (cid, control) \in controls \land (lxng.levelCrossing, cs) \in control.states \\
\land\ lxng \in \mathrm{ran}\, levelcrossings \bullet \forall\, tid : TrainId;\ train : Train;\ ls : LevelState \\
\mid (tid, train) \in control.trains \land (lxng.levelCrossing, ls) \in levelStates \\
\bullet \exists\, s1, s2 : Node \mid (s1, s2) \in train.movingBlock \\
\bullet\ s1 = lxng.levelCrossing \lor s2 = lxng.levelCrossing \\
\land\ lxng.levelCrossing \in control.section \Rightarrow ls.levelState = OCCUPIED \\
\land\ ls.occupiedBy = tid \land ls.barriers = CLOSED \land cs = OCCUPIED \\
\hline
\end{array}$$

ConsistencyAtSwitch as given below.
$NoDerailFR \mathrel{\widehat{=}} NoDerailCFR \land NoDerailAtAll \land ConsistencyAtSwitch$

The first two schemas *NoDerailCFR* and *NoDerailAtAll* are used to prevent derailing where the second one is further composed of two schemas *NoDerailFixed* and *NoDerailAuto* given below. The third one schema *ConsistencyAtSwitch* for checking whether a switch is locked and operated well by the interlocking or manual system is defined as: $NoDerailAtAll \mathrel{\widehat{=}} NoDerailFixed \land NoDerailAuto$.

Derailing may occur, if either a train occupies a track segment of a switch and the state of the switch does not match with the direction of train; or if the state of one of the switches in the moving block does not match with the position and the type of the train. In the schema *NoDerailCFR*, it is specified that if any branch of a switch is in the moving block of a train under a control then control of the switch must be left in the proper direction. A train may derail if one of the switches does not have correct control. In the schema *NoDerailFixed*, it is specified that fixed route train and all switches in the moving block must be controlled by the interlocking. In case of autonomous train, specified by *NoDerailAuto*, it is stated that if the train is autonomous then it is checked that all switches in the moving block of train may be operated by interlocking system or manually.

$$\begin{array}{l}
\textit{NoDerailCFR} \\
\hline
\textit{RISSystem} \\
\hline
\forall\, sw : Switch;\ s : Node;\ ss : SwitchState;\ cid : ControlId;\ control : Control \\
\mid sw \in \mathrm{ran}\, switches \land (s, ss) \in switchStates \land (cid, control) \in controls \\
\land\ sw.switch \in control.section \\
\bullet \forall\, s1, s2 : Node;\ sc : SwitchControl \mid ((s1, s2), sc) \in sw.control \\
\bullet \exists\, tid : TrainId;\ train : Train \mid (tid, train) \in control.trains \\
\bullet\ (s1, s2) \in train.movingBlock \lor (s2, s1) \in train.movingBlock \\
\Rightarrow sc = ss.swControl \\
\hline
\end{array}$$

$$\begin{array}{l}
\textit{NoDerailFixed} \\
\hline
\textit{RISSystem} \\
\hline
\forall\, sw : Switch;\ s : Node;\ ss : SwitchState;\ cid : ControlId;\ control : Control \\
\mid sw \in \mathrm{ran}\, switches \land (s, ss) \in switchStates \land (cid, control) \in controls \\
\land\ sw.switch \in control.section \\
\bullet \forall\, s1, s2 : Node;\ tid : TrainId;\ train : Train;\ sc : SwitchControl \\
\mid (tid, train) \in control.trains \land (s1, s2) \in train.movingBlock \\
\land\ ((s1, s2), sc) \in sw.control \bullet ((s1, s2) \in \mathrm{dom}\, sw.control \\
\Rightarrow sc = ss.swControl \land ss.operation = INTERLOCK) \\
\land\ ((s2, s1) \in \mathrm{dom}\, sw.control \\
\Rightarrow sc = ss.swControl \land ss.operation = INTERLOCK) \\
\hline
\end{array}$$

In the schema *ConsistencyAtSwitch*, it is checked that if a switch is in the moving block of a train then it must be operated well either by interlocking system or manually. At first, critical area of switch is determined, if it is contained in the moving block of any train then switch state and switch operation are verified after knowing the type of train. If a train is of fixed route then the switches in the moving block of train must be interlocked. On the other hand, if a train is an autonomous then a switch in the moving block may be operated manually or may be interlocked. Finally, all the track segments which form stem and both branches of the switch must be occupied in the *states* mapping.

NoDerailAuto

RISSystem

$$\begin{array}{l}
\forall sw : Switch;\ s : Node;\ ss : SwitchState;\ cid : ControlId; \\
control : Control \mid sw \in \operatorname{ran} switches \wedge (s, ss) \in switchStates \\
\wedge\ (cid, control) \in controls \wedge sw.switch \in control.section \\
\bullet \forall s1, s2 : Node;\ tid : TrainId;\ train : Train;\ sc : SwitchControl \\
\mid (tid, train) \in control.trains \wedge (s1, s2) \in train.movingBlock \\
\wedge\ ((s1, s2), sc) \in sw.control \\
\wedge\ ((s1, s2) \in \operatorname{dom} sw.control \wedge (s1, ss) \in switchStates \\
\Rightarrow ss.operation = INTERLOCK \Rightarrow sc = ss.swControl) \\
\wedge\ ((s2, s1) \in \operatorname{dom} sw.control \wedge (s2, ss) \in switchStates \\
\Rightarrow ss.operation = INTERLOCK \Rightarrow sc = ss.swControl)
\end{array}$$

ConsistencyAtSwitch

RISSystem

$$\begin{array}{l}
\forall sw : Switch;\ cid : ControlId;\ control : Control;\ ss : SwitchState;\ s : Node \\
\mid sw \in \operatorname{ran} switches \wedge (cid, control) \in controls \wedge (s, ss) \in switchStates \\
\bullet \forall tid : TrainId;\ train : Train \mid (tid, train) \in control.trains \\
\bullet \exists s1, s2 : Node \mid (s1, s2) \in \{sw.root\} \cup \operatorname{dom} sw.control \\
\bullet \exists s3, s4 : Node \mid (s3, s4) \in train.movingBlock \\
\bullet (s1 = s3 \vee s1 = s4 \vee s2 = s3 \vee s2 = s4) \\
\wedge\ (sw.switch \in control.section \Rightarrow (\exists s : Node;\ ts : TrainInfo \\
\mid (s, ts) \in trackStates \wedge s = s3 \vee s = s4 \bullet ts.trainType = FIXEDROUTE \\
\Rightarrow ss.switchState = OCCUPIED \\
\wedge\ ss.occupiedBy = tid \wedge ss.operation = INTERLOCK \\
\wedge\ (\forall s5, s6 : Node;\ cs : State \mid (s5, s6) \in \{sw.root\} \cup \operatorname{dom} sw.control \\
\wedge\ (s5, cs) e control.states \wedge (s6, cs) e control.states \bullet (cs = OCCUPIED)) \\
\wedge\ ts.trainType = AUTONOMOUS \Rightarrow ss.switchState = OCCUPIED \\
\wedge\ ss.occupiedBy = tid \wedge (ss.operation = INTERLOCK \\
\vee\ ss.operation = MANUAL) \wedge (\forall s5, s6 : Node;\ cs : State \mid (s5, s6) \\
\in \{sw.root\} \vee \operatorname{dom} sw.control \mid (s5, cs) \in control.states \\
\wedge\ (s6, cs) \in control.states \bullet (cs = OCCUPIED \wedge cs = OCCUPIED))))
\end{array}$$

5.2. **Model analysis.** Any specification written in a formal notation does not mean that it is correct, complete and meaningful. The syntax of Z notation is quite complex and contains a number of functions, symbols and key words. The Z/Eves tool was used incrementally in each paragraph for the syntax, type and domain checking of the formal model and immediately was corrected if necessary. By domain checking, it was proved that all expressions appearing in the specification are meaningful. It was observed the domain checking is more difficult than syntax and type checking because the latter is performed automatically while domain checking is performed by interacting with the theorem proving.

Z/Eves was used to perform three types of reductions that are simplification, rewriting and reduction which drastically changed the goal predicate. The reduction commands traverses the current goal, accumulating assumptions and performing reduction on predicates and expressions in the goal. Each of the command can apply certain definitions or theorems to examining the formula and sub-formula of the goal, and may replace it by a logically equivalent formula. The proof by reduce or rewrite repeatedly reduces or rewrites the current goal until it is unchanged. Some schemas were proved with the proof assistance of the tool using such reduction and rewriting techniques.

6. Conclusion.

Both safety and efficiency are primary requirements in design and development of railway interlocking system. In this research, moving block interlocking is selected being the current and future technology increasing efficiency at the railway tracks and formal methods are applied ensuring safety of it. There exists a lot of work on modeling of fixed block interlocking; however, a little work is found on moving block. The work [30] of Simpson is close to ours in which he uses Z, CSP and FDR; however, he does not address the safety issues. An analysis of movement of subway under moving block is discussed for decreasing delay time of the successive trains [22, 23]. Jeong et al. [17] have proposed an algorithm for moving block interlocking using radio signal but this work is only for tracking position of a train. Another related work is presented in [20] using distributed method to schedule the newly added trains in the network. We have described the safety properties for moving block based on the critical components preventing collision and derailing of trains. Initially, we have formalized the abstract safety properties and then redefined by applying the concept of moving block. Further analysis of the properties was done by introducing computer based controls to monitor the trains and network state space.

Z notation is applied because of its rigorous and abstract characteristics. We believe that this experience will be useful to model the other critical systems using the same approach. We observed that the complexity of the system was reduced by decomposing it into its critical components. The use of schema structure facilitated us in reducing complexity because of its abstract and re-useable characteristics. Finally, development from abstraction to detailed analysis made it easy to purpose a simple and understandable model. It is to be noted that this formal model can be applied to any particular interlocking after a further analysis. This is because we have modeled the system and defined the properties based on the requirements of like a real system.

REFERENCES

[1] M. Akillilar and C. Koppe, Increase of effectiveness and efficiency in the interlocking system ESTW L90 5, *Signal und Draht*, vol.100, no.4, pp.26-30, 2008.

[2] T. Akiyama and M. Okushima, Advanced fuzzy traffic controller for urban expressway, *International Journal of Innovative Computing, Information and Control*, vol.2, no.2, pp.339-355, 2006.

[3] M. Banci and A. Fantechi, Geographical versus functional modelling by statecharts of interlocking systems, *Electronic Notes in Theoretical Computer Science*, vol.133, pp.3-19, 2005.

[4] D. Bjorner, *The FME Rail Annotated Rail Bibliography*, Department of Information Technology, Technical University of Denmark, 1998.

[5] J. R. Calame, N. Goga, N. Ioustinova and J. V. D. Pol, Testing of hoorn-kersenboogerd railway interlocking, *Canadian Conference on Electrical and Computer Engineering*, pp.620-623, 2006.

[6] C. Chevillat, C. David, G. Paul, G. Jörn and W. Luke, *Model-Based Generation of Interlocking Controller Software from Control Tables*, Springer-Verlag, Berlin, 2008.

[7] Y. Ding, Y. Bai, F. M. Liu and B. H. Mao, Simulation algorithm for energy-efficient train control under moving block system, *World Congress on Computer Science and Engineering*, pp.498-502, 2009.

[8] M. Fantechi, A. Banci and S. Gnesi, The role of formal methods in developing a distributed railway interlocking system, *FORMS/FORMAT*, 2004.

[9] U. Foschi, M. Giuliani, A. Morzentia, M. Pradella and P. S. Pietro, The role of formal methods in software procurement for railway transportation industry, *Symposium on Formal Methods for Railway Operation and Control Systems*, 2003.

[10] J. Guo, Z. Huang and M. Liu, Research on the railway safety critical system with petri nets, *The 6th International Conference on ITS Telecommunications*, pp.118-121, 2007.

[11] K. M. Hansen, Validation of a railway interlocking model, *Proc. of the 2nd International Symposium of Formal Methods Europe on Industrial Benefit of Formal Methods*, pp.582-601, 1994.

[12] A. J. Harrison and I. D. R. Shannon, The application of formal methods to railway signalling systems specification and the esprit iii project cascade, *The 14th International Conference on Computer Safety, Reliability and Security*, pp.101-112, 1995.

[13] X. Hei and N. Ouyang, The scheduling strategy of concurrent request in distributed railway interlocking system, *An International Journal of Research and Surveys*, vol.2, no.1, 2011.

[14] X. Hei, S. Takahashi and H. Nakamura, Modeling and evaluation of component based distributed railway interlocking system using petri nets, *Research Reports*, Institute of Science and Technology, Nihon University, 2007.

[15] C. Herde, A. Eggers, F. Martin and T. Teige, Analysis of hybrid systems using HySAT, *IEEE the 3rd International Conference on Systems*, pp.196-201, 2008.

[16] A. Janota, Using Z specification for railway interlocking safety, *Transport Engineering*, vol.28, no.1-2, pp.39-53, 2000.

[17] R. G. Jeong, H. S. Cho and S. G. Chung, A study of a new technique for the train tracking system, *The 8th International Conference on Electrical Machines and Systems*, pp.2482-2487, 2005.

[18] S. A. Khan and N. A. Zafar, Towards the formalization of railway interlocking system using Z notation, *Proc. of the 2nd IEEE International Conference on Computer, Control and Communication*, pp.750-755, 2009.

[19] S. A. Khan, N. A. Zafar and F. Ahmad, Petri net modeling of railway crossing system using fuzzy brakes, *International Journal of Physical Sciences*, vol.6, no.14, pp.3389-3397, 2011.

[20] T. Letia, M. Hulea and R. Miron, Distributed scheduling for real-time railway traffic control, *International Multiconference on Computer Science and Information Technology*, pp.679-685, 2006.

[21] M. Lim, Lifecycle-based functional interlocking specification, *Signal und Draht*, vol.100, no.4, pp.7-40, 2008.

[22] F. Lu, M. Song, G. Tian and X. Li, The application of robot formation approach in the control of subway train, *IEEE International Conference on Intelligent Robots and Systems*, pp.5937-5941, 2006.

[23] F. Lu and Z. Zhang, Method for subway operation adjustment based on multi-agent, *IEEE International Conference on Automation and Logistics*, pp.1189-1193, 2009.

[24] M. Matsumoto and B. Keisuke, Application of assurance technology for railway signaling system, *The 8th International Symposium on Autonomous Decentralized Systems*, pp.69-76, 2007.

[25] K. Nakamatsu, Y. Kiuchi, W. Y. Chen and S. L. Chung, Intelligent railway interlocking safety verification based on annotated logic program and its simulator, *International Conference on Networking, Sensing and Control*, pp.694-699, 2004.

[26] M. Peltekova and T. Bonev, Method for modelling real failures in railway system, *The 6th Symposium on Formal Methods for Automation and Safety in Railway and Automotive Systems*, 2007.

[27] M. Ryota, K. Masami and I. Jun, Petri net solver for semiconductor manufacturing plan, *International Journal of Innovative Computing, Information and Control*, vol.4, no.12, pp.3193-3206, 2008.

[28] X. She, Y. Sha, Q. Chen and J. Yang, The application of graphic theory on railway yard interlocking control system, *IEEE Intelligent Vehicles Symposium*, pp.883-887, 2007.

[29] O. Shigeyuki, S. Ryo, K. Miyuki, K. Tadashi, F. Hiroshi and H. Ryuzo, Formal verification of a railway interlocking system by the spin model checker, *Research Reports on Information Science and Electrical Engineering*, Japan, pp.33-38, 2006.

[30] A. Simpson, Towards the mechanical verification of moving block signalling systems, *Technical Report*, Oxford Brookes University, 1999.

[31] J. M. Spivey, *The Z Notation: A Reference Manual*, Prentice Hall, 1992.

[32] H. Takeuchi, C. J. Goodman and S. Sone, Moving block signalling dynamics: Performance measures and re-starting queued electric trains, *IEE Proc. of Electric Power Applications*, vol.150, no.4, 2003.

[33] X. Hei, T. Sei, N. Hideo, F. Mitsuyoshi, L. Koji and K. Sato, Improving reliability of railway interlocking system with component-based technology, *The Journal of Reliability Engineering Association of Japan*, vol.28, no.8, pp.557-568, 2006.
[34] N. A. Zafar, Modeling and formal specification of automated train control system using Z notation, *IEEE International Multi-Topic Conference*, pp.438-443, 2006.
[35] N. A. Zafar, Formal specification and validation of railway network components using Z notation, *Software, IET*, vol.3, no.4, pp.312-320, 2009.
[36] N. A. Zafar and K. Araki, Formalizing safety properties preventing collisions and derailing for moving block railway interlocking system using VDM-SL, *Asian Journal of Information Technology*, vol.3, no.5, pp.313-328, 2004.