The assessment of interlocking control algorithm developed by Rhapsody

Yong-Ki Yoon, Won-Young Kim, Yong-Kyu Kim
Train Control Research Team, Signaling & Electrical Engineering Dep., Korea Railroad
Research Institute, Uiwang City, Gyounggi do, Korea

Abstract
Train signal system is responsible for safety of train operation by controlling train safety distance and train route. In Europe, new safety standards are firmly defined for verifying and validating safety of this system (EN50126, EN50128). These standards require certification about organization, process, tools which are necessary for system development. In case of developing main algorithms of the signal system, the requirement for these standards can be achieved by applying the development tools of formal method based on Unified Modeling Language (UML). This paper indicates the result of assessment of train interlocking control algorithm using a simulator which realizes train operation based on moving block. And the algorithm based on moving block was designed by Rhapsody, the software development tool.

1. Introduction
The development of railway systems starts with the creation of a system requirements specification. System requirements specifications, including those for safety related systems, are mostly written in natural language and completed by diagrams. These have to be formal and they have to be adequate for describing the properties of railway signaling systems. A fundamental property of the railway system is that it is possible to regard its parts as objects. From these objects then new and more specialized objects can be derived and some of them can communicate with each other. This leads to the obvious conclusion of using an object-oriented (OO) approach for specifying those systems, thus implying object-oriented means of description. Unified Modeling Language (UML) is the choice. Its means of description are based on graphics and are easy to understand. However, the UML description tools are for the most part semi-formal. If in a certain range of application the necessity arises then a proper formal back-up has to follow so that necessary steps of verifying the safety become valid. Since the Unified Modeling Language (UML) is on its way to become a standard among the object-oriented specification languages. The system components can be defined as objects or classes of the model. The relationships between these objects then depict the relationships between the system components. The object oriented model precisely, compactly and comprehensibly represents the static structure and dynamical behavior of the system. Another advantage of this model is the ability to transform it into a formally verifiable form and by employing model checking it can be verified that safety requirements are met. Consequently, a formal verification can already be conducted during the definition phase[1].
Rhapsody constitutes the core part of the Unified Modeling Language. Rhapsody embarked on an effort to develop an integrated set of diagrammatic languages for object modeling, built around statecharts, and to construct a supporting tool that produces a fully executable model and allows automatic code synthesis. Object-model diagrams specify system structure by identifying object classes and their multiplicities, object relationships and roles, and subclassing relationships. Statecharts describe system behavior. A statechart attached to a class specifies all behavioral aspects of the objects in that class. In addition, Rhapsody support message sequence charts, also called sequence diagrams, as a reflective language (which captures parts of the thinking that goes into building the model—behavior included—or is used to derive and present views of the model to aid analysis). Message sequence charts describe the possible ways a system behaves in terms of scenarios or use cases[2].
As shown Fig. 1, small section of a network contains two sets of points, P1 and P2. In addition, there are three trains currently in the locality: T1, T2, and T3. Each of these trains is traveling in
The trackside system is required to continuously inform each train how far in advance of its current position it is safe to travel. This information is determined by criteria such as the position of other trains, and the current state of the network's sets of points. The fundamental hazards associated with a moving block system are collisions and derailments. At the highest level, the safety requirements associated with moving block systems are consistent with those associated with fixed block systems: to ensure that no two trains can occupy the same section of track and that points are suitably locked and aligned for a train traveling over them[3].

When apply the concept of moving block system to interlocking system, the characteristics of interlocking system can be represented as follows;

(a) Closed loop control: the interlocking system constitutes a closed loop of communicating continual information between trackside and train. Because an interlocking systems in a way of track circuits can't able to send information from train to trackside, safety is ensured by using approach and stick locking, setting a safety margin(time). The interlocking system can communicate necessary information of train caused by constitution of the closed loop, it does not set the hour for controlling the train route.

(b) Admitting multi train approach: train route consists of several blocks. Each block has information about train ID, course of train and state of blocks and so on. Unlike a block of track circuit, it can make train route with one block. The running track in the station yard of several blocks is able to make several routes. Thus, several trains (railway cars) can go into a running track.

(c) Optimizing train operation: the interlocking system allocates train number (ID) and train running direction to the train route. In case a train enters the different track by random errors, this system can handle the error by comparing the established information (train number, direction) of the track and those of the train. Existing system operators can recognize the sort of trains and operations, and then can set courses, but these systems cannot recognize the error about setting courses by limited information (route occupation, course setting)[4].

2. Organization of advanced interlocking system

As you see in Fig.2, the train control system of ground and on-board system controls the distance and route of train with radio communications[5].

The on-board system consists of train distance control unit and train route control unit. Each part has separated radio equipment. The on-board train interlocking control unit inspects the equipped transponder on running track, and judges an approach or advance in the section of train interlocking control. It determines whether to precede operation of radio equipments or not. The ground train interlocking control unit identifies an approach advance from the part of train interlocking control by exclusive radio sets on ground and on-board. The part of ground train interlocking control detects position of train by using radio devices between ground and on-board control. The train interlocking control part manages track switches and signals which are facilities on job-site based on information of approach and advance in the section of train interlocking control. They are derived from the information about train position through the train distance control part and delivered by way of radio communication through the on-board train route control part.

In site equipment, there are antennas which communicate with radio transmission parts of trains, and ground transponders. Site control equipments consist of a processing unit, a radio
transmission unit, and I/O units.

![Diagram of Train Control System configuration](image)

Transponders, as the equipments to identify trains’ coming and going on tracks, transfer to the on-board train separation control parts/devices the information that trains pass on the transponders when the conditions of trains are detected by transponder antenna. The train separation control unit sends the information about the estimated transponders and running directions of trains, to the radio transmission unit of the site control equipment via the on-board radio transmission unit. Based on the information transferred to the processing unit via the ground radio transmission unit, the field control equipments can estimate the condition of tracks occupied or passed by trains.

The radio transmission unit plays a role of receiving and then sending the data transferred from the train route control parts of trains to the processing unit parts. The I/O unit controls point machines and signal posts in compliance with the commands come from the processing unit. It also performs the function to transfer the state data of signal posts and point machines to the processing unit.

![Diagram of Field Site configuration](image)

3. Designing interlocking process and model of the advanced interlocking system

1) Interlocking process of smart train interlocking process

The interlocking process of the interlocking system is equal to interlocking processes allowing train’s approach after searching its route in the existing train interlocking systems based on track circuits.

(a) a requested train route

A train route means that a train can run from a starting point to an arrival point, and a train route is requested by automatic or manual mode on a control panel. Information for the requested route is as follows, and unlike the existing system, it contains a wide variety of information.

(b) searching a train route and permitting train approach

Searching a train route in the train route control system by the route requirement of the control panel creates the information about states of route and block. Information about states of route
are variable contents applied to current route, those of block manage fixed contents such as contact relation with block and track switch. Process of requirements about track course in the train route control system is as follows.

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Route requirement</td>
<td>Response for required route</td>
</tr>
<tr>
<td>Train ID</td>
<td>Train ID</td>
</tr>
<tr>
<td>Route ID</td>
<td>Route ID</td>
</tr>
<tr>
<td>Train course</td>
<td>Route state</td>
</tr>
</tbody>
</table>

Table 1: I/O data of advanced interlocking system

<table>
<thead>
<tr>
<th>Phase</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Identifying the existence of route ID</td>
</tr>
<tr>
<td>2</td>
<td>Formation of route state table (recording train ID)</td>
</tr>
<tr>
<td>3</td>
<td>Identifying the existence of any obstacles or deadlocks in the established route</td>
</tr>
<tr>
<td>4</td>
<td>Selecting blocks available for running</td>
</tr>
<tr>
<td></td>
<td>Identifying a need of conversion in the corresponding track switch</td>
</tr>
<tr>
<td>5</td>
<td>Converting the track switch</td>
</tr>
<tr>
<td>6</td>
<td>Interlocking the track switch and transmitting block line</td>
</tr>
<tr>
<td></td>
<td>(applying to train distance control)</td>
</tr>
</tbody>
</table>

Table 2: Searching process of a train route

2) Modeling the advanced interlocking system

We developed the interlocking control algorithm using Rhapsody, S/W development tool, based on the above interlocking process. In order to verify interlocking control algorithm, it is required to prepare various verification scenarios and to apply them to interlocking control algorithm. Consequently, we drew up Domain Overview composed of ISLogicPkg, InterfacePkg and HardwarePk.

A package represents a logical grouping of elements. These elements may be anything; they are classes and objects. A domain is an independent subject matter with its own vocabulary and object model, such as the ISLogicPkg, ther InterfacePkg and the HardwarePk. Package may have interface. A blue dotted line in following Fig. 4 represents the interface between the HardwarePk and the InterfacePkg. There exists also the interface between the ISLogicPkg and the InterfacePkg although it hasn’t been represented in the Fig. 4.

Fig. 4: Domain Overview of the interlocking system
The HardwarePkg functions as a tester that generates various event conditions required to verify the ISLogicPkg. As shown in Figure 5, the HardwarePkg drew up 8 events. Instead of connecting the HardwarePkg and the ISLogicPkg directly, we had the HardwarePkg go through the InterfacePkg. It means that we can easily replace the HardwarePkg by another hardware package in the future without having to modify the ISLogicPkg.

The Interlocking control algorithm development is done at the ISLogicPkg. The ISLogicPkg consists of 10 classes and 4 object model diagrams. Classes composing each diagram are as follows. Fig. 6 shows the DataManager Overview Diagram for the ISLogicPkg in table 3 with 6 classes, Logic, DataManager, SegmentDB, RouteDB, SignalDB and SwitchDB.

<table>
<thead>
<tr>
<th>Diagram</th>
<th>classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataManager Overview</td>
<td>Logic, DataManager, SegmentDB, RouteDB, SignalDB, SwitchDB</td>
</tr>
<tr>
<td>Logic Overview</td>
<td>Logic</td>
</tr>
<tr>
<td>Logic Data Overview</td>
<td>Logic, DataManager, StateManager</td>
</tr>
<tr>
<td>StateManager Overview</td>
<td>Logic, StateManager, RouteState, RouteStateID, SwitchState</td>
</tr>
</tbody>
</table>

Table 3 : Diagram and classes of the ISLogicPkg

As described in the Table 3 above, classes are classified into logic part, state part, data and data manager part. As a result, the Logic class undertaking interlocking is composed of statechart as simple as shown in following Fig.7. Statechart describes both how objects communicate and collaborate and how they carry out their own internal behavior. Sequence diagram shows scenarios of messages exchanges between roles played by objects. This functionality of Rhapsody can be used in numerous ways, including analysis and design scenarios, execution traces, expected behavior in test cases, and so on.
Fig. 8 shows the sequence diagram for the LogicData Overview Diagram. The sequence diagram is a design of LogicData scenarios composed of Logic, DataManager andStateManager. During the development process, sequence diagrams are used for the following primary purposes:

- In the early system requirements phase, they are used for use case description.
- In the implementation phase, they verify that all conditions are met in terms of communication between classes.
- In the testing phase, they capture the actual system trace.

Therefore, there is a need to facilitate comparison between sequence diagrams because they should be identical. The execution message sequence should match the specification message sequence.

Fig. 8 depicts the comparison of two sequence diagrams. The left is animation results of the LogicData Overview Diagram based on random event conditions and the right is the results based on the development specification of the LogicData Overview Diagram. After comparing those two, it is proved that there are many differences on StateManager.

In this case, we need to determine whether the specification is inaccurate or an error exists in the implementation. In both cases, we should correct the modeling error and then repeat the testing cycle to determine whether we have fixed the problem.

3) The assessment of the interlocking system using a simulator

Rhapsody is for the most part semi-formal. So, the necessity arises a proper formal back-up has to follow so that necessary steps of verifying the safety become valid. The simulator shown in Fig. 9 below is used to verify developed interlocking control algorithm.

Moving block system controls train route by using real-time train location information, rather than using track circuit type train occupation information used in existing interlocking device. In
order to verify the interlocking device, therefore, we implement a simulator that calculates train movement and speed by using rotation signal generated due to speed control via micro motor and performs track determination, train spacing control, route control and train simulator and interlocking process. By using this simulator, we validated that interlocking control algorithm performs its own functions well.

![Photograph of the simulator](image1)

**Fig. 9 : Photograph of the simulator**

![Test results of the interlocking control algorithm using the simulator](image2)

**Fig.10 : Test results of the interlocking control algorithm using the simulator**

4. Conclusion

We have developed the algorithm for advanced interlocking system based on moving block. For developing this algorithm, we have used Rhapsody which is a software development tool under UML, and identified functionality of the developed algorithm by applying sequence diagram.

Also, we have evaluated the interlocking algorithm developed by a simulator which can realize the train operation based on interlocking. Through all these processes, we have identified functionality of the advanced interlocking system such as route controls for two trains on the same route.

In the process of evaluation, we have found several problems of the simulator, and could not identify whether the advanced interlocking algorithm contributes to optimize train operation. So, we plan to evaluate the interlocking algorithm in terms of optimization for developed train operation by improving function and performance of the simulator.

References


