Design and implementation of a distributed railway signalling simulator

X. Hei[1,2], W. Ma[1], L. Wang[1] & N. Ouyang[1]
*[1]School of Computer Science and Engineering,
Xi'an University of Technology, China
[2]State Key Laboratory of Rail Traffic Control and Safety
(Beijing Jiaotong University), China*

Abstract

The Distributed Railway Signalling System (DRSS) is a new signalling system, in which all devices including trains, switch point and signals, as well as position checking, interact and exchange information based on some logic constraint relations. These devices operate independently to ensure train safety. Based on this idea, we have presented the concept of modelling these device actions with G-nets (an Object-oriented Petri Net tool) in Comprail 2006. In this paper, a simulation system that we developed is introduced in order to conduct experiments on DRSS and verify its feasibility. The simulator is based on the concept of DRSS and includes mainly six classes and their functionality modules: station layout automatic generation, train operation, position checking, switch point and signal. In addition, the instance generation of all classes and timetable design are considered in the simulator. It is possible to verify and simulate almost all functions with this simulator, such as train protection, route process, interlocking logic verification and terminal device procedure, etc.
Keywords: distributed railway signalling system, simulator, object-oriented.

1 Introduction

A railway signalling system has been developed over the long history of railways, and has been vital in ensuring the safe operation of trains. However, computers have been used in such safety-critical systems for no longer than 30 years [1], and they have demonstrated a high level of safety and reliability. One drawback of the existing computerized railway signalling systems, however, is that they require the development of different software for different stations, which tends to introduce unreliable human errors. Further, they are difficult to upgrade due to their lack of standardization, both in hardware and software. A possible solution to overcome these problems is to apply modular-based technology to railway signalling systems. For this, a novel system named Distributed Railway Signalling System (DRSS) was presented in Comprail 2006 [2]. In the new signalling system, all devices including trains, switch point and signals, as well as position checking, interact and exchange information based on some logic constraint relations. These devices operate independently to ensure train safety.

It is vital to design the logic functions and control flows for such a safety-critical system. A convenient approach is to develop a simulator based on the designed logic and control flows.

In this paper, a simulation system we developed is introduced in order to conduct experiments and verification on DRSS. The simulator is based on object-oriented concept and includes mainly six classes and their functionality modules: information display module, initialization module, train operation, position checking, switch point, and signal. Also the instance generation of all classes, timetable design are considered in the simulator. It is possible to verify and simulate almost all functions with this simulator, such as train protection, route process, interlocking logic verification and terminal device procedure.

2 Distributed Railway Signalling System (DRSS)

Compare with traditional signalling system, DRSS needs not the centralized computer for controlling the whole system. All terminal devices work independently and exchange messages via network to ensure safety operation of trains. In the case of a typical interlocking system, these devices include signals, points and track units. Signals indicate whether the train can run or not by displaying green or red. Points are devices for controlling turnouts which determine the direction in which trains move. Track units detect whether or not there is a train on the track. If there is, then other trains are prohibited from entering this section of track until the first train leaves.

Consider an interlocking system in a station, the architecture and message interaction flows of DRSS are illustrated in Figure 1. All devices are composed of logic process part and action part. The logic process part receives/sends messages from/to other devices, and makes decision. The action part provides mechanical output according to the orders sent by logic process part, such as displaying green or red for a signal device, turning over the switch for a point.

The development process deals with standardized hardware and software for the interlocking devices. Control flows of the interlocking devices are based on their function specification.

The development strategy of the DRSS is shown in Figure 2. Structure of standardized devices consists of hardware part and software part. The hardware

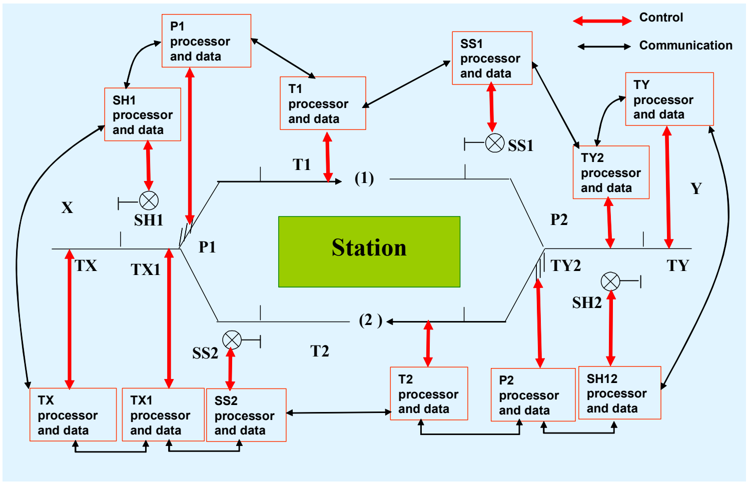

Figure 1: Architecture of the DRSS in the case of an interlocking system.

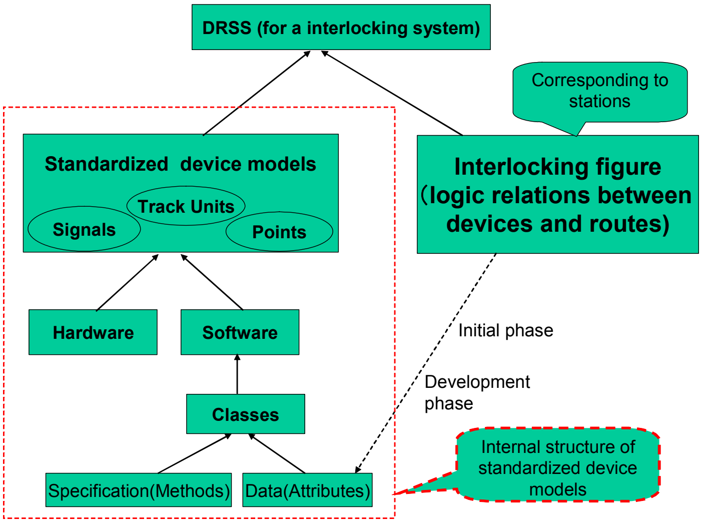
DRSS (for a interlocking system)
Corresponding to stations
Standardized device models
Track Units
Signals
Points
Interlocking figure (logic relations between devices and routes)
Hardware
Software
Initial phase
Development phase
Classes
Internal structure of standardized device models
Specification(Methods)
Data(Attributes)

Figure 2: Development strategy of the DRSS.

specifications include device board design, CPU, digital circuit, input/output, etc. The software specifications include the necessary modules design of typical interlocking devices. Each module is similar with a class or object which inherits from one kind of device class. The device control flows are expressed by methods, while interlocking logic data related to a specific station are expressed by attributes.

When the devices are initialized, the logic data will be loaded into the devices, and then the devices operate based on these data.

Once the devices have been verified safe enough, they can be ordered and produced when a new station is constructed. What engineers just need to do is analyzing the logic relations and allocating some basic attributes such as device ID to each device.

3 DRSS simulator design

Authors have proposed a Petri net-based designing approach for DRSS in reference [3]. Toward development of the DRSS simulator, object-oriented methodology is an ideal choice. Thereby UML tools like sequence diagram, class diagram and activity diagram are used for the system design [4, 5].

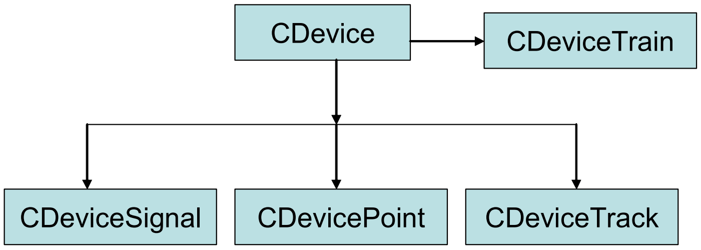

Figure 3: Device classes and their inheritance relations.

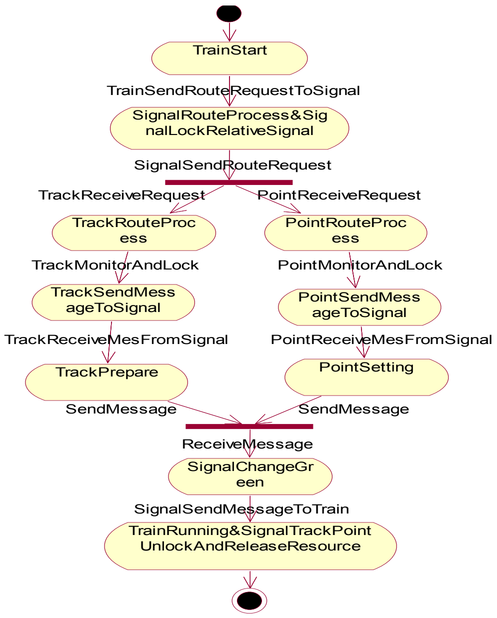

Figure 4: Activity diagram when a train is approaching a signal.

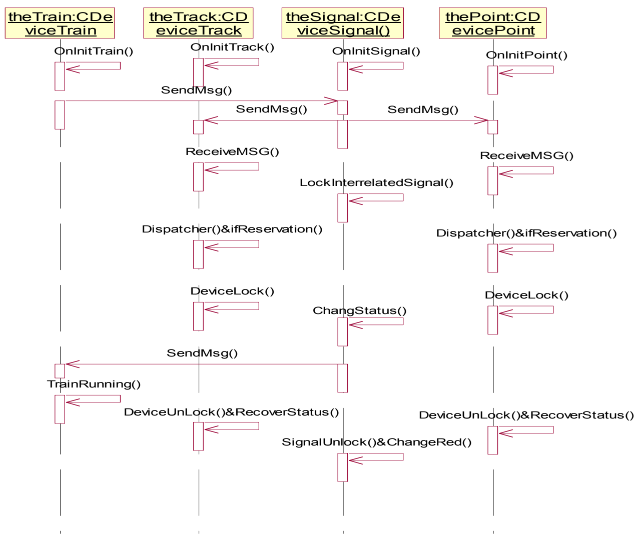

Figure 5: Sequence diagram of the four classes and objects.

In the DRSS simulator, there are four device classes and two function classes are designed. Device classes include *CDeviceSignal*, *CDevicePoint*, *CDeviceTrack* and *CDeviceTrain*, which are shown in Figure 3. These four device classes inherit from *CDevice* class. Function classes include station layout automatic generation and message. Based on these classes, the instance generation of all classes and timetable design are considered in the simulator.

The process when a train comes can be depicted as activity diagrams of these devices. With moving of the train, a signal will start the route reservation process and request to lock the conflict signals. All devices which are related to the requested route check their states and send a response message to the signal. If and only if all these devices are ready for this route request, the route can be reserved and the signal displays green. Figure 4 gives the activity diagram. These procedures for sending and receiving messages and actions of each device are predefined as member functions of class.

The sequence diagram is designed as Figure 5.

4 Implementation of DRSS simulator

4.1 System flowchart and modules

There are mainly six modules in the simulator: information display module, initialization module, train module, signal module, switch point module and track module.

1) Information display module: displaying system information, train information, timetable window.

2) Initialization module: initializing communication module, signal module, switch point module, track module based on the interlocking interlocking logic relations, and initializing train module with train schedule information.
3) Signal module: description of the associated signal, point and track, displaying green or red based on the associated devices.
4) Switch point module: locking/unlocking point, setting position (Normal or reverse) of switch point, indicating the reachable stations when the switch point is in normal or reverse position.
5) Track module: determining whether the train is on some track segment or not.
6) Train module: description the train ID, status (running or stop), the train starting station and destination, and all pass through stations, the current track section and the next, train acceleration and deceleration function.

The system process flowchart is illustrated in Figure 6.

4.2 Displaying stations and railway lines

For the implementation, an important step is to display the station layout automatically. Consider the universal property, the station layout has to be record as data and a drawing module is needed. The module can be divided into five steps.

i. Dividing the main window's client area (size 800 × 500) into 32 × 20 grids, each grid is 25 pixels long side.

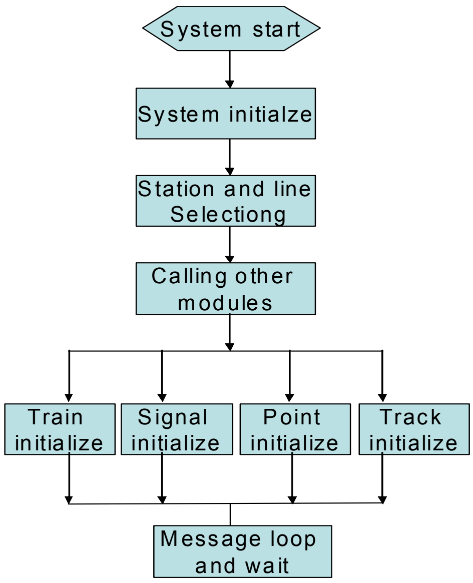

Figure 6: System process flowchart.

ii. Constructing an array with 21 rows × 33 columns, the array subscript (i, j) is corresponding to the main window's client area coordinates (j * 25, i * 25), where i = 0, 1 ... 20, j = 0,1 ... 32.

iii. The element value of the array is assigned to an equipment ID or connection mark of two adjacent rail sections when there is a device. Otherwise, the element value is assigned to 0 which indicates that there is not equipment.

iv. In the array, equipment ID should be one of three kinds: switch machine ID, track circuit (including the station platform) and signals. Here switch machine's ID is assigned 301-399; track circuit ID is 101-199 (track segment) and 201-299 (station platform), and signal ID is 401-499.

v. Designing a switch machine table. The fields include switch machine ID, device ID, device ID, device ID, which means that the switch machine is associated with the three equipments followed.

vi. Creating data for each device and drawing these devices based on the data table described above, when the station and lines need to be introduced.

With the drawing module above, it is convenient to display a different station by designing a corresponding two dimension array. It executes when the simulator initializes and a station and line is selected.

4.3 Layout of the simulator

The window is divided into four areas: main client area, system information area, train information area and timetable area, as shown in Figure 7. Station and

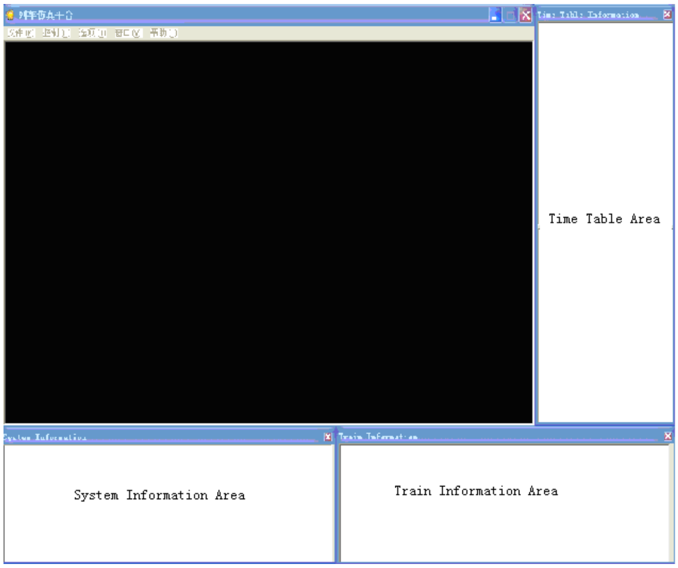

Figure 7: Main window of the simulator.

railway lines will be displayed in the main client area. System running information will be displayed in the system information area. Operation information of all trains will be displayed in the train information area. Timetable area lists the departure as well as pass through time of all trains from the first–run to the last-run.

5 Conclusion

The simulator shows the essential features of the DRSS: all device classes process all messages and make decision independently. This process starts with approach event of train. The DRSS simulator provides a platform for almost all experiments and analysis, including exploring the effect of device amount on message process, communication protocols design, etc. In addition, the stochastic failures or events can be inserted into the operation process of trains. This work will be carried out soon. Therefore, it is possible to verify control and schedule logics and simulate almost all functions with this simulator, such as train protection logic, route process logic as well as logic verification and terminal device procedure.

Acknowledgement

This work is supported by the State Key Laboratory of Rail Traffic Control and Safety (Contract No. RCS2008K008) and Natural Science Basic Research Plan of Shaanxi Province (2009JQ8010).

References

[1] K. Akita, T. Watanabe., H. Nakamura., I. Okumura: “Computerized Interlocking System for Railway Signaling Control; SMILE”. IEEE Trans., May 1985: Ind., 1A-21.
[2] X. Hei, H. Mochizuki, S. Takahashi. & H. Nakamura: “Modeling distributed railway interlocking system with object-oriented petri-net”. In 10th International Conference on Computer System Design and Operation in the Railway and Other Transit System, Prague, Czech Republic, 2006, pp.309-318.
[3] Xinhong Hei, Sei Takahashi, Hideo Nakamura,: Modelling and Analyzing Component-based Distributed Railway Interlocking System with Petri Nets, Institute of Electrical Engineers of Japan (IEEJ) Transactions on Industry, Sec. D, Vol.129 , No.5, 2009.5.
[4] Object Management Group, Unified Modeling Language Specification v.2.0, www.uml.org, September 2003.
[5] C. Lindemann, A. Thummler, A. Klemm, M. Lohmann, and O. Waldhorst: Performance Analysis of Time-enhanced UML Diagrams Based on Stochastic Processes, In Proc. of the 3rd Workshop on Software and Performance (WOSP), pp. 25–34, Rome, Italy, 2002.