Research on the simulation of an Automatic Train over speed Protection driver-machine interface based on Model Driven Architecture

B. Y. Guo, W. Du & Y. J. Mao
State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, China

Abstract

The principle of Model Driven Architecture (MDA) was drawn into the emulation research of an Automatic Train over speed Protection (ATP) driver-machine interface (DMI). To realize the functions of the DMI, a new method based on the MDA principle was raised. Specific to the requirement of the DMI, the ICV (Core Interface-Frame Controller-View) model was proposed. This is the Platform Independent Model of the ATP driver-machine interface. ICV is a View-centred GUI model that includes a Core Interface and a Frame Controller. The View was used for the description of interface visualization. The Frame Controller accomplished the communication between the driver and the on-board vital computer (VC) by the display of different views. The Core Interface provided the information bridge among View, the driver and VC. Then the detailed transform rules from the Platform Independent Model to the Platform Specific Model were drawn up. The transform rules were separated into two parts. One part realized the core communication function to ensure the accuracy of the system communication interface by the auto-transform method and, according to the definition of the Platform Independent Model, the other part built each module of the ICV model using manual mode. The ultimate complete ATP driver-machine interface system satisfied the emulation requirements, and has been used for the research of the evaluation and testing on the CTCS-3.

Keywords: ATP driver-machine interface, MDA, GUI model, simulation.

WIT Transactions on The Built Environment, Vol 114, © 2010 WIT Press
www.witpress.com, ISSN 1743-3509 (on-line)
doi:10.2495/CR100021

1 Introduction

The control of operation signalling for Railway China has developed from manual operation by drivers, who follow the traffic command of ground signals, to automatic speed control by the Train Controlling System, which receives the information sent from the ground [1]. The ATP (Automatic Train over speed Protection) driver-machine interface is displayed at the centre of a LCD monitor, which is configured with a speaker and a keyboard. Drivers are notified with all kinds of information about the train and status of the ATP by sound and graphical information, and then are able to change its working mode and status by input through the keyboard. As a media of displaying the train information and speed command, the human-machine interface is the only interface to communicate with the backend train running control system; it plays an important role in the running process of the train as its normal display affects the arrival time and safety of the train.

The CTCS-3 simulation and testing platform is a research platform hosted by the National Key Laboratory of Rail traffic Control and Security, Beijing Jiaotong University, in order to make researches on systems and solutions, and evaluate the equipment testing for the CTCS-3. This system includes the train security computer, track information receiving unit, transponder information receiving unit, speed sensor, human-machine interface and 3D scene, to simulate the train running environment to be as real as possible. The simulation of the ATP driver-machine interface has a great significance in the implementation of a simulation platform of the entire train control system.

The principle of Model Driven Architecture (MDA) was drawn into the emulation research of the ATP driver-machine interface (DMI). To realize the functions of the DMI, a new method based on the MDA principle was raised. Specific to the requirements of the DMI, the ICV (Core Interface-Frame Controller-View) model was proposed. Then the detailed transform rules from the Platform Independent Model to the Platform Specific Model were drawn up. The ultimate complete ATP driver-machine interface system satisfied the emulation requirements, and has been used for the research of the evaluation and testing on the CTCS-3.

2 Simulation method of the ATP driver-machine interface based on MDA

MDA is the collection of a series of Standards (MOF, UML, CWM and XMI) and Technology (CORBA, Java, C++, etc.), which are the basis for supporting MDA [2]. The core idea of MDA is to form a CIM (Computation Independent Model) based on users' needs, including the development purposes, performance and requirements of the software to be developed in the system development process [3]. According to the CIM model, using the above standards and technology, the platform-independent, highly summarized models are abstracted and concrete, which are called PIMs (Platform Independent Models). Then, the

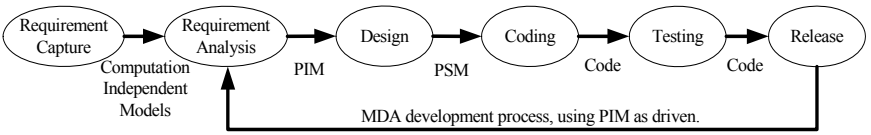

Figure 1: MDA software development cycle.

transform rules are defined based on the specific implementing technology platform. The PIM is transformed to the PSM through the defined transformation rules and tools, and the PSM will be converted into executable code automatically.

The development process of the MDA-based system is shown in Fig. 1.

Using MDA, the system development process is detached from the building process of two models: one is the establishment of the PIM; the other is the establishment of the PSM, and the key technology is the conversion between the PIM and the PSM. In the beginning of the system development, the PIM should be established, which is independent of the specific implementation technology and platform and is the high-level abstraction of the system. Then, according to the transformation definition, the PIM is converted into the PSM, which is closely related with the specific implementation technology and platform. The framework of the MDA includes the PIM, PIM description language, transformation rules, PSM, PSM description language and several other elements [4]. In traditional software development processes, the model represents not only the demand, but also the realization of specific technologies. Using MDA, models are classified into PIM, representing demand, and PSM, representing the realization of specific technologies, and therefore, the demand and technologies are related.

To develop an ATP driver-machine interface simulation system, the requirement must be analyzed above all, and the function of the ATP driver-machine interface could be described using UML. On top of this, the PIM was established, which did not contain any platform-related details. In the description of the PIM, the demand should be summarized and summed up; restraint describing language should be used to achieve the transformation. After the completion of the PIM description, the PIM was mapped to a particular simulation development platform, and then the PSM was obtained. The PIM-to-PSM transformation rules were divided into two parts. One part is to carry out the communication function, since the core function of the ATP driver-machine interface is to accomplish communication with the Vehicle Computer [5, 6]. In order to ensure the accuracy of communication, the automatic conversion mode was used in this part [7]. The other part is to complete the construction of the modules and interaction among the modules according to the definition of the ICV model. The manual mode was used in this part. The PSM was generated by a combination of these two conversion methods and then the ultimate simulation system was completed.

The comparison between the MMI development framework and the MDA development framework is shown below.

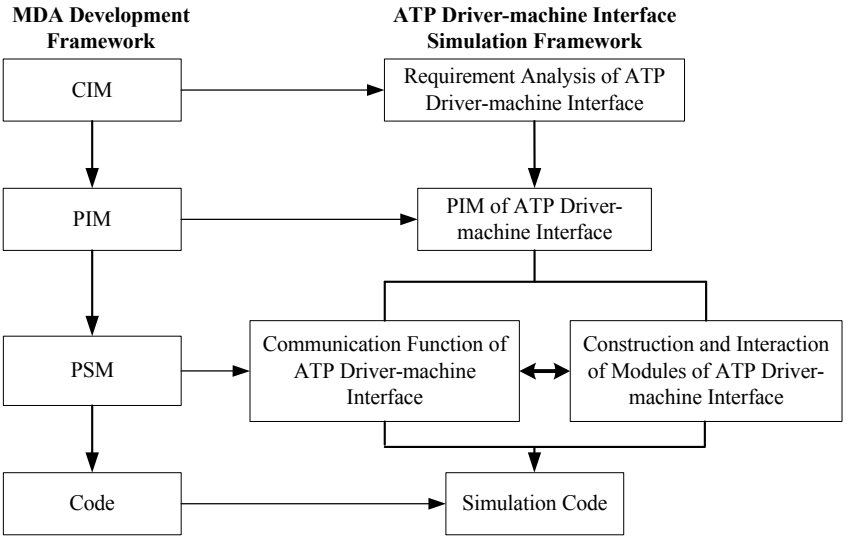

Figure 2: Development framework of the DMI simulation system.

3 Requirement analysis of the ATP driver-machine interface simulation

The Vehicle Computer sent information to the ATP driver-machine interface at a fixed frequency. The ATP translated the information to a readable data, according to the prior agreed rules, and showed the data on the interface as certain rules. The ATP followed multi-level hierarchical design ideas and was decomposed into various views in its logic functions. Each view could be divided into multiple sub-views and each sub-view was a further decomposition of its parent view. Drivers may make driving operations based on the information displayed in all levels of views, including entering data, such as Driver ID, train number, train length and so on, controlling the train independently for functions such as mitigation and change the running status, and responding to the information sent by the vehicle security computer, such as the need for confirmation of the driver when transforming from CTCS-2 to CTCS-3. The information would be sent back to the Vehicle Computer by the ATP after the driver finishes the operation and then responses were sent back to the ATP after being confirmed by the Vehicle Computer, which formed a closed loop for the information communication.

The train initial data, such as Driver ID, train number, train length and so on, are input at the first step when the driver started to drive. Then the start button was pressed. The driver should drive in accordance with the interface display. During driving, the information transmitted from the Vehicle Computer was responded to by the driver and the driver could control the train independently.

The main responding operations are confirmation of operation level and the status of the front track.

From the workflow, the functions and use case of the system were confirmed, including: 1. data display; 2. data input; 3. mode selection; 4. carrier frequency mode selection; 5. selection and confirmation of operating level; 6. release selection; 7. departure selection; 8. driver response. The ATP driver-machine interface shows the information sent by the Vehicle Computer and contacts the Vehicle Computer and the train driver. Therefore, it can be determined that the driver and Vehicle Computer are system participants. The system use case diagram is shown in Fig. 3.

4 Establishment of the PIM for the ATP driver-machine interface

The ATP driver-machine interface is a graphical user interface. To build the PIM of the ATP driver-machine interface based on MDA is to design graphical user interface models at the system point of view. In this research, combined with an

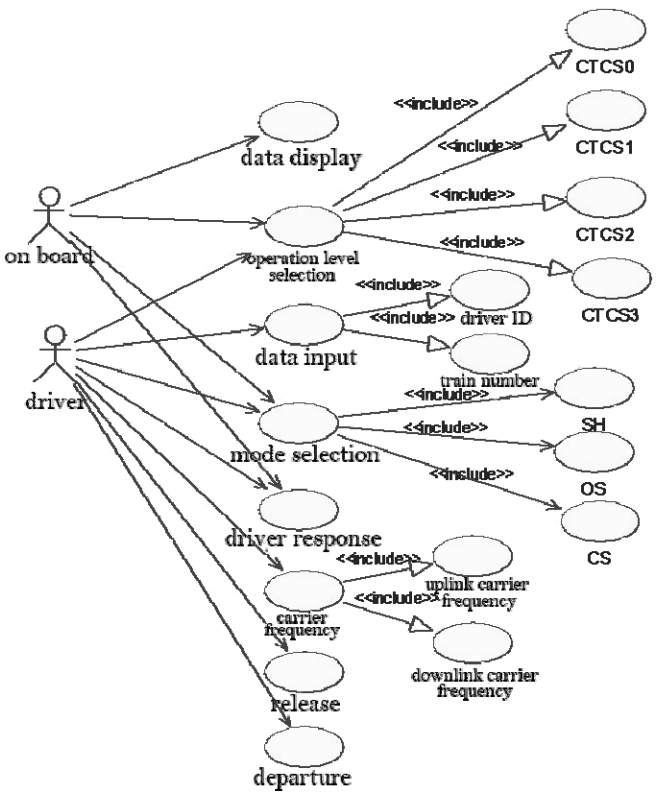

Figure 3: Diagram of the ATP driver-machine interface.

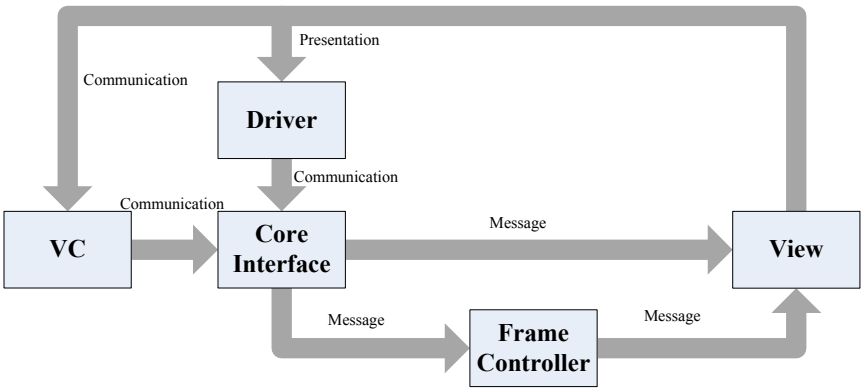

Figure 4: ICV model.

important GUI model, the Seeheim model [8], a kind of PIM for the ATP driver-machine interface was presented. That was the ICV, the Core Interface-Frame Controller-View model. The ICV was a kind of GUI model whose centre was View, including the Core Interface and Frame Controller. The visible part of the user interface was described by View, and the tasks from the driver and vehicle computer were accomplished by the frame controller through each view. The core interface was used to offer an information exchanging interface for train the driver and the Vehicle Computer. The model is illustrated in Fig 4.

Since multi-level hierarchical design ideas were used in the ATP driver-machine interface, the View decomposed the interface into various views in its logic functions. Each view could be divided into multiple sub-views and each sub-view was a further decomposition of its parent view, but the sub-views did not have to be called by parent views, while some shortcut keys were set. Those frequently used sub-views would be called by shortcut keys rather than by parent views, and this facilitated the driver's operation. The static characteristics of the View included size, location and its own form of property, while the dynamic behaviours of the View included the internal action and communication between the View and Vehicle Computer. The View is the core of the ICV model. The View of each level could fulfil its specific function. The hierarchical and modular description of the complex ATP driver-machine interface could be actualized by the use of the View module.

The information response was as a core in modelling the View mode. The View dealt with the messages from the Vehicle Computer and driver by the information response process, for example, the current speed display and calling the sub-views. The sub-views of each level in the View mode interacted in-house. Several Views of level 2 and Views of level 3 had the ability of sending information. According to the incoming control information, the corresponding information was sent to the Vehicle Computer.

The button information collection process in the model was integrated into the driver module. Since its primary role was to capture the driver's button information and send it to the ATP driver-computer interface by the way of

communication, this part was dissociated in the periphery of the ATP driver-computer interface model and it was not necessary to build a separate module for this part. The "Driver" would be representing this process instead in the model.

The Core Interface module had a dual mission. One part was to receive the information from the Vehicle Computer and the other was to receive the driver's button information. It provided an interface between the vehicle equipment and the train driver and established a buffer zone. As a result, the efficiency and maintainability of the code have improved.

The main function of the Frame Controller was to receive control information from the Core Interface, which is responsible for switch scheduling among each view and to control the operation of each view. This module was divided into two parts. One part was used to receive the control information from driver and open the appropriate view according to the driver's manipulation. The other part was used to receive the control information from the Vehicle Computer and open the appropriate view according to the incoming message.

5 PSM construction of the ATP driver-computer interface

When the PIM was constructed, the transformation work from the PIM to the PSM could be launched. Because the current transformation tools can only accurately converse the elements of the PIM into the PSM elements, it was necessary to manually complete the construction of the PSM to achieve specific functionality. According to the past practice, the research was divided into two parts and each part of work is as follows.

The first part was to mainly complete the communication function, including the following aspects.

1. The elements in the ATPInterface Class of Core Interface, which is the core one in the PIM, should be converted into the corresponding function's elements of the PSM.

2. The operations in the ATPInterface Class of Core Interface, which is the core one in the PIM, should be converted into the corresponding operations of the PSM.

3. The elements in the SendInformation Class of the View in the PIM, which need to communicate with the Vehicle Computer, should be converted into the corresponding function's elements of the PSM.

4. The operations in the SendInformation Class of the View class in the PIM, which need to communicate with the Vehicle Computer, should be converted into the corresponding operations of the PSM.

This part is the emphasis of the ATP driver-machine interface simulation, using Rational Rose to automatically generate codes to complete communication between the Vehicle Computer and the ATP driver-computer interface.

The tasks to be done in the other part are as follows.

1. Construct the PSM according to each divided class in the PIM.

There were three parts of the PIM: the Core Interface, Frame Controller and View. The Core Interface contained two sub-modules: one is in charge of receiving information from the Vehicle Computer; the other is in charge of

receiving information from the driver. There are also two sub-modules in the Frame Controller. One is responsible for accomplishing the control on the View from the Vehicle Computer, and the other is to accomplish the control on the View from the rail driver. The View module was divided into several sub-modules according to its interface and all these sub-modules were classified in accordance with different view parts. Each sub-module should be described while the PSM is constructed.

2. Information exchange among the PSMs should be completed according to the interaction among the various modules in the PIM.

Information exchange is the top priority of the ATP driver-machine interface simulation. After the construction of the PSM, interaction among the various models should be accomplished. The Core Interface module had to achieve the interaction between the Vehicle Computer and the ATP driver- machine interface and interaction between the driver and the ATP driver-machine interface. Then the information coming from these two areas were divided into two parts: one was the data message, which would be sent to the View module to display; the other was the controlling information, which will be sent to the Frame Controller to complete the task of calling for the View. The View received data information from the Core Interface and controlling information from the Frame Controller to display the view.

The graphical programming language called G language was used in this part. The PSM description was carried through in the LabVIEW platform. Finally, the simulation system would be finished.

6 ATP driver-machine interface running a typical example

The simulation platform of the ATP driver- machine interface is part of the Rail Control System Simulation Platform in the lab, and has been used for system research, program research and equipment testing and evaluation on the CTCS-3.

Fig. 5 shows the ATP driver-machine interface operation chart. In the picture, it can be seen that the ATP driver-machine interface was shown in the centre of the driving device's LCD screen. Speakers and keypad are around the screen,

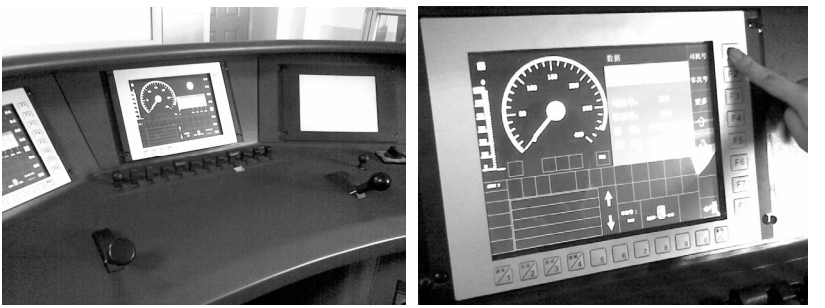

Figure 5: ATP driver-machine interface operation chart.

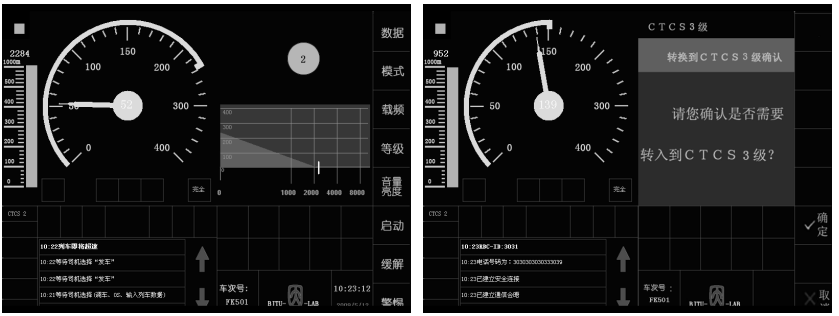

Figure 6: ATP driver-machine interface operation status and the transforming confirmation menu to the CTCS-3.

and the driver can input some relevant information through the keyboard to change the mode of the ATP system and work status. The ATP driver-machine interface can correctly display various status images. This is shown in Fig. 6.

7 Conclusion

The first step of this method was to do a requirement analysis, following which the PIM can be established. Combined with the GUI model, a new model called the ICV (Core Interface-Frame Controller-View) model was proposed as the PIM of the ATP driver-machine interface simulation. The process of transforming from the PIM to the PSM was divided into two parts. One part is to finish the most important section of the ATP driver-machine interface, communication, especially the communication between the Vehicle Computer and the ATP driver-machine interface. The other part was used to establish the PSM according to each module of the PIM and the interaction among the modules. Then the system simulation would be completed. It has been proven that this method saves development time and enhances the portability and accuracy of the system. The ATP driver-machine interface simulation system has been used for the research of evaluation and testing on the CTCS-3.

Acknowledgement

This work is supported by the Key Science and Technology Research Project of the Chinese Ministry of Education (No. 109010).

References

[1] Wang Xi, Tang Tao. Design and Realization of Train Operation Control System Onboard MMI Based on UML. Journal of System Simulation, 18(2), pp. 338-361, 2006.
[2] Heng Xiangan. Research on the Modeling and Simulation Method based on MDA. Changsha: National University of Defense Technology. 2005.

[3] Jishnu Mukerji. Model Driven Architecture-A Technical Perspective. www.omg.org/cgi-bin/doc/ormsc/2001-07-01.
[4] Jiang Chun. MDA Method and MDA Modeling Based on UML. Journal of Shenyang Institute of Engineering: natural Science, 4(1), pp. 67-93, 2008.
[5] Gronbaek, J, Madsen, T.K, Schwefel H.P. Safe Wireless Communication Solution for Driver Machine Interface for Train Control Systems. System, 4, pp. 208-213, 2008.
[6] Ceccarelli A., Majzik I., Lovino D. A Resilient SIL 2 Driver Machine Interface for Train Control Systems. Dependability of Computer, 6, pp. 365-374, 2008.
[7] Deuk Kyu Kum, Soo Dong Kim. A Method to Generate C# Code from MDA/PSM for Enterprise Architecture. Computer and Information Science, 6, pp. 238-243, 2006.
[8] Sun Xiaoping, GUO Tengchong, WEI Mingzhu, etc. A UML-Based Object-Oriented Graphic User Interface Design Model. Computer Science, 30(5), pp. 108-112, 2003.