Qt-Based Cross-platform Design of Management System for Distributed Real-time Simulation Platform

Xibao Wang[a], Ge Li and Peng Wang[b]

College of Information System and Management, National University of Defense Technology, Changsha 410000, China;

[a]xbwang1990@126.com, [b]wangpeng_nudt@nudt.edu.cn

Keywords: Cross-platform technology, distributed real-time simulation platform, management system, user interface, QMainWindow, QTreeWidget.

Abstract. At present, with the development of cross-platform software technology and the cross-platform application gaining more and more attention, there emerged many multi-platform support, superior performance, cross-platform software development tools and integrated development environments, such as Qt Creator. In order to run the distributed real-time simulation platform on different operating systems, such as Windows, Linux and Mac OS, this paper tries to design and implement the management system user interface for distributed real-time simulation platform using the cross-platform framework based on Qt. Firstly, this paper builds the main frame window of management system through inheriting the QMainWindow class. Then the paper designs the system view, tree widget and central component respectively. Finally, this paper assembles the widgets appropriately into the main frame. All above work lays a good foundation for the follow-up system development and improvement.

Introduction

Simulation software is a kind of special software for simulation applications, its functions include model specification and processing (i.e. modeling), simulation experiment implementation and control (i.e. run control), analysis, display and documentation of data and results (i.e. results processing) and storage and management of knowledge, data, graphics or model (i.e. database). According to these functions, simulation software can be divided into simulation package, simulation language and simulation platform. Simulation platform as a middleware facing problems and users between operating system and system applications occupies a vital position in the field of simulation technology. [1]

Since the 1970s, China began to introduce, transplant and develop simulation software. The fact is that a large number of professional simulation applications and software have been developed, but with the rapid development of computer hardware network connectivity and communication all these software are facing with the coexisting problems of how to adapt to heterogeneous network and multi operating systems.

Currently, due to Microsoft Windows operating system occupies an absolute advantage in the PCs, most applications and software which are based on Microsoft MFC framework and libraries cannot run on other heterogeneous operating systems. If we abandon the original developments to redevelop those applications and software, we will not only spend a lot of money and manpower, but also waste the mature and excellent software achievements. However, if we develop these applications making them support cross-platform at the beginning, we can not only reduce the development cost remaining all the functions and features of software under the new operating system platform, but also improve the deficiencies of the original software and add new demands to them, thereby achieving the goal of improving software quality and extending the software life cycle. In addition, as each country all around the world attaches great importance to national information security, along with the gradual deepening and promotion of the domestic operating system, domestic operating system is in urgent need of a large number of application software to enrich itself platform

capabilities, especially the applications involving the aerospace simulation, combat simulation and other military simulation software related to national strategic security.

This paper introduces the cross-platform design and development based on Qt Creator about the distributed real-time simulation platform management system. Qt as the current popular, cross-platform C++ GUI application development framework, it supports most desktop operating systems, such as Linux, Windows and Mac OS, of course some embedded operating systems. Because applications written in C++ using Qt has a good cross-platform feature, they can run on a variety of operating systems without changing the source code, which greatly shorten the development cycle and reduce the development costs.

Background Knowledge

Introduction of KD-DRT Simulation Platform. The distributed real-time simulation platform consists of a database server, some design and simulation computers, a management computer and other compositions. KD-DRT is such a support platform for multi-parallel distributed real-time simulation which bases on database, distributed real-time simulation engine, real-time synchronization and other modules. KD-DRT is an open set of system-level hardware and software integration solutions, which supports distributed real-time simulation test configuration and operation management. Its software module consists of test configuration, run-time, semi-physical I/O, simulation process management and monitoring, data analysis and display (see Fig 1). [2] In this paper, the distributed real-time simulation management system corresponds to the test configuration module of KD-DRT Simulation Platform, which includes model management, node management and test configuration, which are very important to the simulation platform.

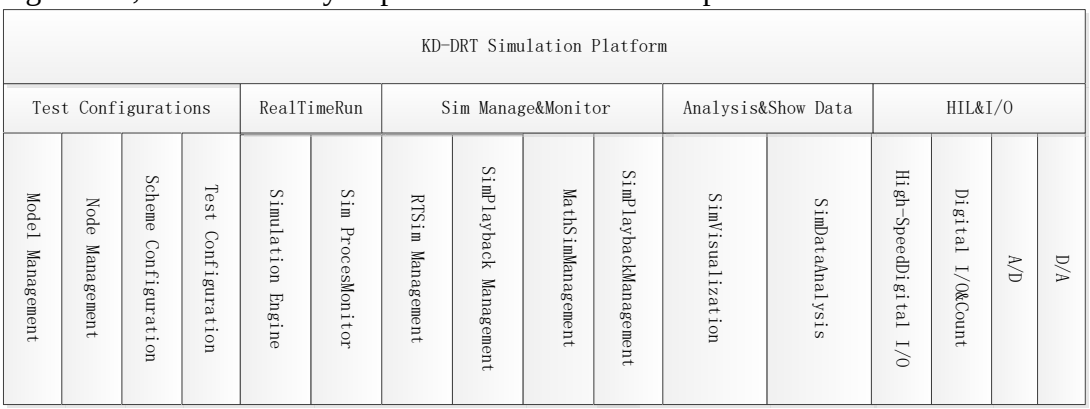

Fig. 1 The composition of KD-DRT software module

Introduction of Cross-platform Technology. Cross-platform refers to that the programming language, application software or hardware devices can work on varieties of operating system platform (such as Linux, Windows and Mac OS). Generally, cross-platform software development is divided into object code level and source code level. The former is known as Java language based on Java virtual machine, whose characteristic is “development once, run anywhere”. Since the java program cannot be executed directly by the operating system without the Java virtual machine, which in return leads to low running speed of the application. [3] Moreover, the Java virtual machine itself does not have the cross-platform ability, so the cross-platform technology based on Java machine is incomplete. Another cross-platform approach is based on cross-platform class library. It makes use of a kind of common computer language supported by most operating systems, such as C++, to write program, and then to compile and link to generate the object code, finally achieve the same performance, known as “write once, compile anywhere”. As a result, the cross-platform approach based on source code can not only solve the performance problems of the applications, but also take full advantage of the operating system characteristics. The latter one is a complete and effective cross-platform technology.

Class library is a specially designed program package of code and data used to solve some specific problems. Cross-platform class library tries to abstract and unify the access to window system, such as window components, widgets, input, event processing, multi-thread processing, file I/O, network

and 2D/3D graphics, which achieves transparent realization on each target system. Qt is such a cross-platform class library, whose function builds upon the underlying APIs of different platforms (see Fig 2). [4] In Qt class library, all widgets are drawn through Qt Designer Tool. Developers can also extend and customize their own widgets by overloading the supplied virtual function.

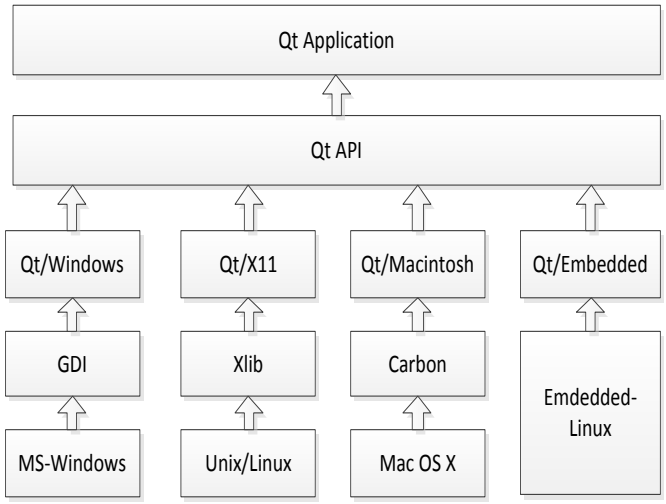

Fig. 2 The architecture of Qt class library

GUI Design of Management System. The management system for the distributed real-time simulation platform is an important part of the KD-DRT simulation platform, which is responsible for test configuration and management. Simulation management system stores the simulation test program data, system model data and computer node data needed during the simulation process. Through the user interface users can manipulate the data stored in the database, such as add, delete, modify and refresh data items.

Design of System Main User Interface. As a cross-platform graphic user interface application framework, Qt provides all required functionality for application developers to establish all kinds of graphic user interface. QMainWindow class provides the developers with an application main window (see Fig 3), including a menu bar, a tool bar, multiple dock widgets, a status bar and a central widget. [5] The developers can easily and quickly build their own custom main window frame through inheriting and overloading the supplied class. In view of the characteristics of distributed real-time simulation management system and the similarities of the majority of GUI application development, this paper adopts the provided and convenient main form, namely the QMainWindow class, to design and build the user interface main frame of the simulation management system.

Typically, there are two methods for developers to build their application interfaces using Qt Creator. Pure code implementation is one method; another method is to draw UI implementation using the Qt Designer Tool. Although the latter method is intuitive, it hides many details and essence of the realizations. Taking into account the future expansion and maintenance of the system, this paper takes advantage of both methods to build the main user interface framework.

This paper follows the common steps to build the application graphic user interface using the Qt Creator. Firstly, this paper creates main framework through inheriting the QMainWindow class, and then uses codes to implement the menu bar, tool bar, status bar and dock component with setting their properties respectively in the derived class – ControlSystemWindow (see Fig 4). The central widget is a custom widget derived from the QWidget class drawing with the Qt Designer Tool. Next, this paper will concentrate on the system view design based on the dock widget using as a widget container. The system view includes three sub-views, the project outline sub-view, with each child view organizing and managing the data information in tree structure. The project outline child view provides different test scenarios according to the user demand; the model management child view is responsible for providing various mathematical models for simulation; the node management child view determines the system architecture.

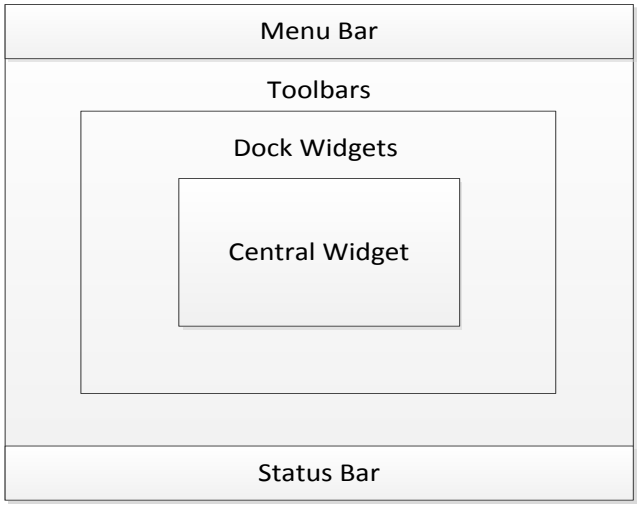

Fig. 3 Typical schematic of QMainWindow class

In order to ensure the entire interface being more concise and compact, this paper assigns the QTabWidget instance to arrange the three child views, such as the model management child view, with each tab page holding each child view. We just need to call the addTab() method to add the custom tree widget to each tag page. The users can switch between the three child views by clicking on each tab page. Then this paper calls the setWidget() method to put the QTabWidget instance into the dock widget, which can stop at different positions over the main frame. Finally, this paper completes the building of the system view by calling the addDockWidget() method to insert the dock widget into the main window frame.

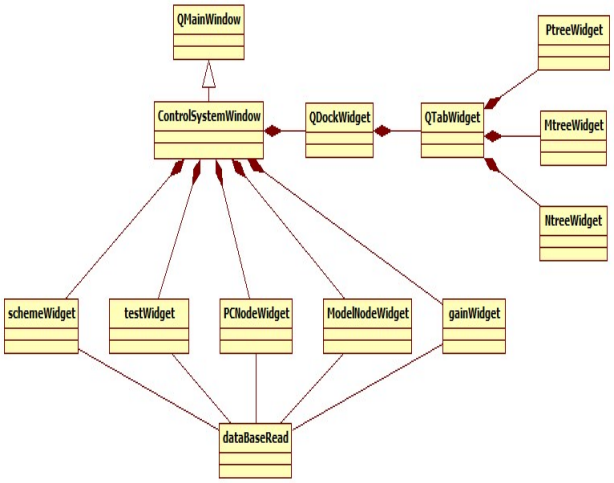

Fig. 4 Class diagram of main user interface

Design of the Model Management Sub-view. Model management child view as a significant part of the system view not only displays the system model with different functions for users, but also allows users to add node, delete node, drag node spanning the nodes in the tree widget and other series of operations in the tree view. Moreover, each node in the tree view not only responds to mouse double-click events, but also provides right-click menu function. This paper customizes a tree widget as the model management child view by deriving the convenient view class – QTreeWidget, which is based on model/view framework of Qt, to meet the functional requirement of simulation management system. The QTreeWidget class defines a hierarchical tree model within itself and displays the tree model in the tree style. Data items are represented as QTreeWidgetItem within the model. As long as the designers create QTreeWidgetItem instance and specify the parent node, QTreeWidget will automatically record and maintain the parent –child relationships of all nodes to form a complete tree model. [6] In this paper the customized MtreeWidget class (see Fig 5) achieves right-click menu, double-click, drag and drop functions respectively by overriding the pure virtual functions of base class – QWidget, extending the QTreeWidget class functions.

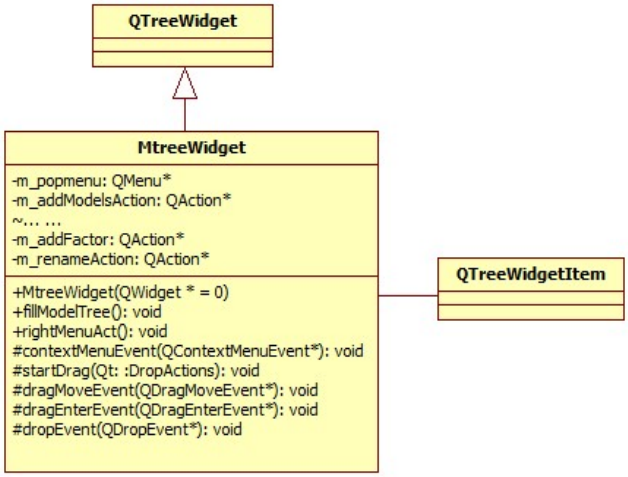

Fig. 5 Class diagram of model management sub-view

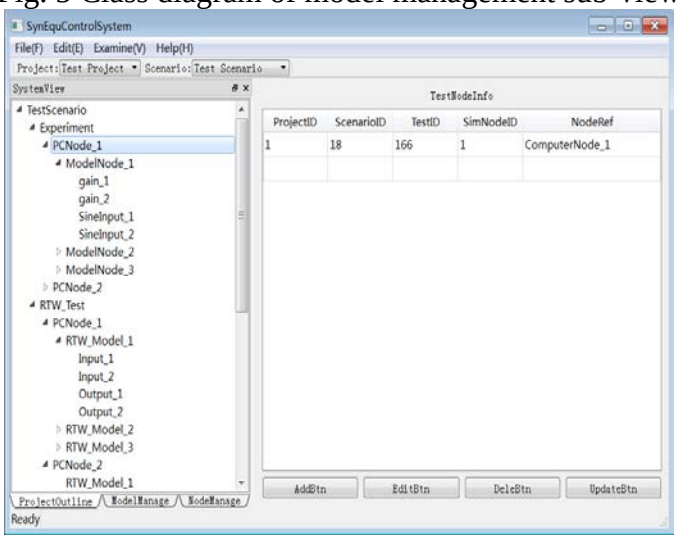

Fig. 6 GUI of the distributed real-time simulation management system

Summary

In this paper, we elaborate the composition and the main frame of the distributed real-time simulation management system user interface and realize a schematic design and implementation of the main user interface by using the cross-platform Qt Creator integrated development environment. Particularly, we focus on illustrating the design of model management child view as a representative. As a part work of the research project in regard to cross-platform transplantation technology, I will further deepen the researches on cross-platform transplantation of the distributed real-time simulation management system. The following work will transfer to the design and migration of the database from MS-SQL Server to MySQL and ultimately realize the cross-platform transplantation from multi-level for the simulation management system. I hold that in the coming future the cross-platform technology research work to enable the software with the same functionality run on different operating system platform will gain more and more attention and the achievements will be more fruitful.

References

[1] Kedi Huang, Technology of System Simulation, first ed., National University of Defense Technology Press, Changsha, 1998.

[2] Xinyu Yao: submitted to Journal of Computer Simulation Technology (2009).

[3] Chaosheng Xu and Wei Shi: submitted to Journal of Sci-tech Information Development and Economy (2006).

[4] Yanmin Li and Yueming Lei: submitted to Journal of Communication and Information Technology(2009).

[5] Di Wu, Zero-based Programming with Qt4, first ed., BeiHang University Press, Beijing, 2010.

[6] Wenzhou Lu, Development and Programming with Qt5, first ed., Publishing House of Electronics Industry, Beijing, 2014.