Design and Implementation of an Emergency Traffic Evacuation Management Platform for Urban Areas Based on Eclipse RCP

Wenwen Kang
Dongguan Research Institute of CASIA
Cloud Computing Center, Chinese Academy of Sciences
Dongguan, China
The State Key Laboratory of Management and Control for Complex Systems
Institute of Automation, Chinese Academy of Sciences
Beijing, China
kangwenwen@casc.ac.cn

Fenghua Zhu, Yisheng Lv, Gang Xiong
Beijing Engineering Research Center of Intelligent Systems and Technology
Institute of Automation, Chinese Academy of Sciences
Beijing, China

Li Xie
Department of Information Science & Electronic Engineering
Zhejiang University
Hangzhou, China

Bin Xi
School of Information Science and Technology
Xiamen University
Xiamen, China

***Abstract*—An Emergency Traffic Evacuation Management Platform is developed in this paper to support evacuation planning and decision making for city managers. The platform is based on Eclipse RCP, a programming framework kept updating by the Eclipse RCP open source project. Architecture design and implementation of the platform are mainly discussed. Modularization is the basic idea when building the platform. At the last of the paper, we suppose that a sudden disaster has happened somewhere in Hangzhou and the platform is used to manage the evacuation planning. Experiments show that evacuation time increases linearly with the amount of evacuation demand in a certain range, and we draw the conclusion that balanced improvements on the whole road network should be made instead of only on the "bottlenecks".**

Keywords—Emergency Evacuation Management; Heuristic Route Selection; Eclipse RCP

I. INTRODUCTION

China is located on the Pacific Rim in Southeast Asia where natural disasters happen most frequently in the world. It is among few countries that are attacked by most types of disasters and suffers the most every year. Along with China's rapid development of economy and accelerating process of urbanization, urban population booms, resulting in frequent man-made accidents of all kinds. We are hearing more and more report about toxic gas leakage, fires and terrorist attacks and so on. These accidents make a big threat to peoples' lives and properties. City governments are putting more and more concern on how to handle emergency evacuation efficiently during these accidents by the support of researchers' studies [1].

Under this background, we propose to build an Emergency Traffic Evacuation Management Platform (ETEMP) in this paper. With this platform's help, researchers get a convenient approach to studying emergency traffic evacuation. Evacuation managers of city governments can also get scientific and intelligent decision supports to help planning evacuations and save peoples' lives and properties to the most.

II. DESIGN AND IMPLEMENTATION

A. Architecture of the Platform

The platform is designed to be an executive application under the windows operating system. Following the MVC (Model, View and Controller) modeling technique [2], all the functionalities of ETEMP are parted into modules [3]. Architecture of the platform is shown in Fig.1. The platform is mainly composed of five modules. They are Resource Management Module, Map Editing Module, Parameter Setting Module, Data Collecting Module and Results Exploring Module. Details about these modules are as follows.

- Resource Management Module: This module is used to manage the resources. We categorize all the resources into five classes according to their levels. They are relatively called working space, project, directory and file in decreasing order of level. The higher level resources could contain one or more lower level resources. The working space is the highest level. All resources are organized in projects and all projects are stored in a single working space. Each project is corresponding to an evacuation task. Not like directory and file, working space and project are only logical

concepts and have no corresponding entities stored in dist. This module provides user interfaces to create and edit resources when necessary, and broadcasts resource changed events to interested modules.

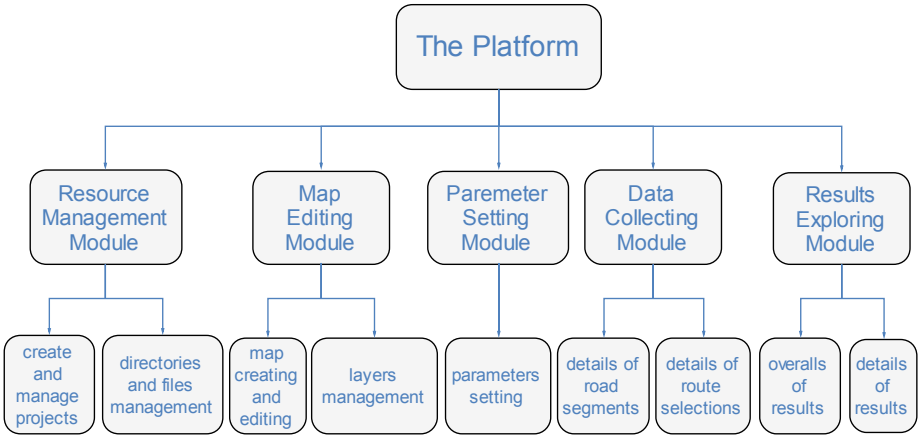

Fig.1. Architecture of the Platform: Module Division

- Map Editing Module: This module is used to create and edit maps when necessary. Map data is stored in specifically formatted files. There are a few digital map formats. We chose Shapefile format from the Esri (Environmental Systems Research Institute, America) [4] in our case because it is provided with open source software packages that support reading and writing on it [5]. Theoretically a perfect Map Editing Module should not only enable users to create and edit maps, but also extract geographical elements from map data. To achieve this goal, a lot of specialist knowledge is needed about presentation, storing and synchronization of digital map data. This is beyond our ability. For simplicity, we don't implement all the functions. Instead enough work has been down to enable uses to directly import Shapefile-formatted digital map data into the platform for further use.

- Parameter Setting Module: This module is used to set parameters of an evacuation task. Parameters play an important role to the task performance. A little change of a parameter may make a big difference to the final results. This module enables users to study emergency evacuation under different conditions only by changing the parameters though this module. This is very useful when conducting sensitivity analysis where a large series of experiments under different conditions are needed.

- Data Collecting Module: This module is used to automatically collect data produced by a task. In the process of an evacuation task, it always produces a lot of data, among which most are useful for further study. This module makes the collecting work automatic.

- Results Exploring Module: This module is used to explore the results. This module processes analyses of the data obtained by the Data Collecting Module and produces the final results in a readable way. Besides, it presents the results in a visual way for users.

B. Implementation Framework: Eclipse RCP

Today software requirements are changing all the time. To survive under the frequent changes, a large-scale client application must have the properties of modularization, cross-platform and native impressions. If developers build an enterprising application from the ground, they must be equipped with specialist developing skills and rich developing experiences. But not all developers have this ability. Luckily, Eclipse RCP is a basic framework having all these properties, which makes it possible for developers to build a rich client platform quickly without having all that skills and experiences. Note that under the Eclipse RCP framework, the graphical user interfaces are based on SWT/JFace library [6], which makes native impression possible.

You may not hear about Eclipse RCP if you are not a java developer. The term Eclipse RCP is short for Eclipse Rich Client Platform. It's a brilliant framework aimed to provide developers a platform to quickly build a large-scale client applications which supported by the underlying hardware [7]. There exist a lot of successful and brilliant examples based on Eclipse RCP, including the "IBM Lotus Workplace" used to enterprise management and problem analysis from the IBM Company, and "Maestro" used to do space experiments from the NASA [8]. Eclipse RCP is becoming more and more popular among developers which we decide to choose as the implementation framework of our platform.

Following the architecture of the platform designed in Fig.1, we developed several components in Java programming language. They are the menu bar, the toolbar, the status bar, the navigation view, the layer view, the welcome view, the map editor, the results exploring editor, the road details editor, the route selection editor and the preference page. Among these components what you may not be familiar with are the views and editors. The "view" and "editor" are two basic window elements in Eclipse RCP which enable the interaction between user and the application. From their name we know, the "view" is mainly aimed at representation while the "editor" at data manipulation.

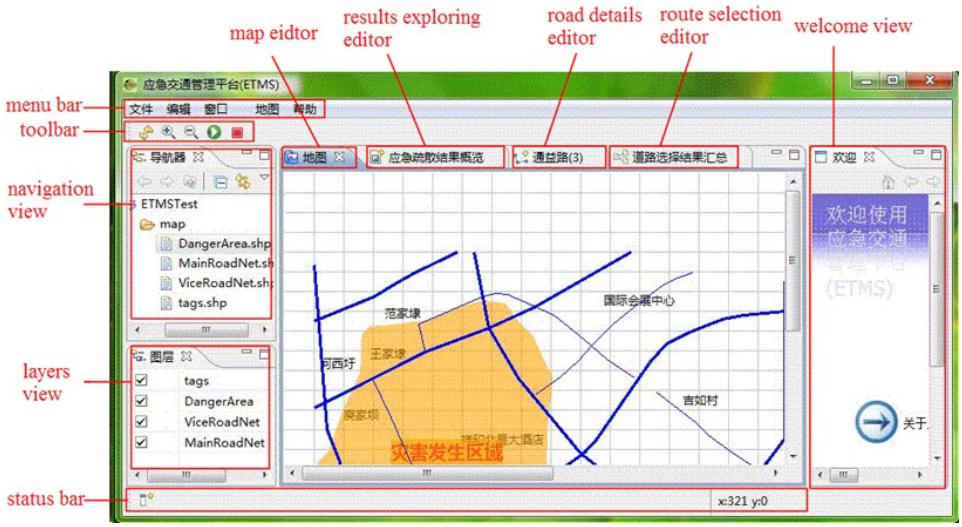

Fig.2. Appearance of the Platform

The final appearance of the platform is shown in Fig.2. On the top of the main window are the menu bar and toolbar, and the status bar is located at the bottom, like all other PC applications. The body area of the main window is divided into three parts. At the left top is the navigation view where projects in current working space are navigated. At the left bottom is the layer view where layers of the map are listed. You can toggle the visibility of the layers through a simple click on the checkboxes. In the middle all editors are arranged in a tap control. You can view the map imported when the map editor is activated. On the right is the welcome view which provides a quick start of the platform for new users.

Details of the programming work are beyond the scope of this paper. References [5]-[10] provide enough information about this if you are interested.

III. AN EXAMPLE OF A CERTAIN AREA IN HANGZHOU

In this section, we take a certain area in Hangzhou as example. We imagine that a sudden disaster of some kind happened in the area. This example produces reasonable results and proves the effectiveness of the platform.

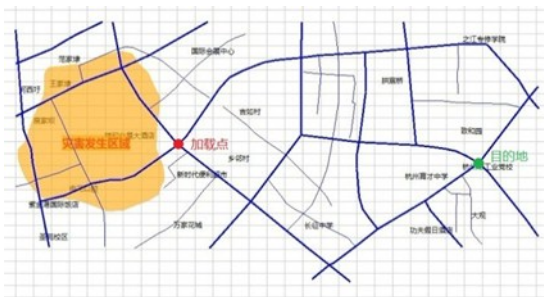

Fig.3. Actual Road Network of an Area in Hangzhou

The actual road network is represented by Fig.3, and the orange (or gray if in black-white mode) area denotes where affected by the disaster. Primary evacuation routes in this area are denoted by thicker lines. The target of the evacuation is to send people and their properties in the hazard area from the loading point which is denoted by the red solid dot (the left one) to the safe destination which is denoted by the green solid dot (the right one). Fig.4 is the topology map of the actual road network in Fig.3. Each road segment is identified by a number. Note that the road segment is unidirectional if there is an arrow on it.

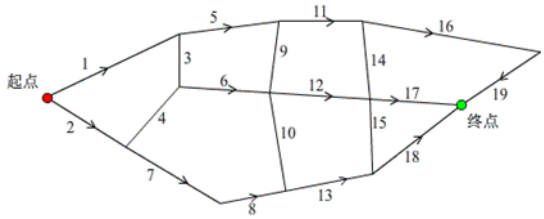

Fig.4. Topology Map of the Actual Road Network

Fig.5. Results Exploring Editor

Run the platform implemented in section II, create a new project (evacuation task) and import the digital map of the area and other resources needed. After setting proper parameters, run the task by clicking on the “run” button in the menu bar or toolbar. Wait until the process finishes and you will see the results in an editor like Fig.5. In this editor we can see how long the evacuation lasts, how many vehicles are evacuated, and the minimum and maximum time it takes to evacuate a vehicle. Besides, there are hyperlinks to each road details listed in the lower area. By clicking a hyperlink, the details of the corresponding road segment are represented in a new editor, as

shown in Fig.6. The chart you see in the road details editor is a component provided by an open source project called JFreeChart [11].

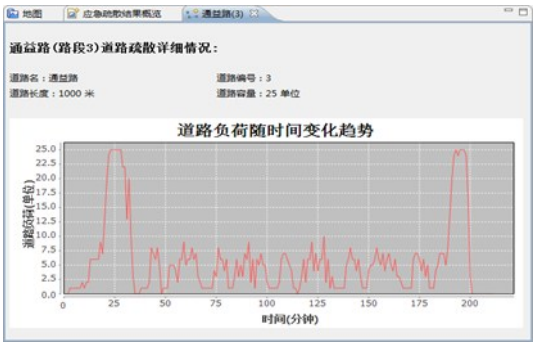

Fig.6. Road Detail Editor of Tongyi Road

Furthermore, we can get evacuation time by changing the total amount of evacuation demand. The result is shown in TABLE.I. By fitting the two series in Matlab we find that they have linear relationship which can be illustrated by the following equation.

$$T = 0.13Q - 28 \tag{1}$$

In (1), T denotes evacuation time in unit of minute, and Q denotes the total amount of evacuation demand (number of vehicles in this case).

TABLE.I. AMOUNT OF EVACUATION DEMAND V.S. EVACUATION TIME

Amount of Demand	900	1200	1500	1800	2100	2400
Evacuation Time(min)	91	132	175	211	249	293

During the evacuation, some roads are nearly free while others are terribly congested as a result of myopic decisions by the drivers. It is these congested roads acting as bottlenecks that delay the whole evacuation. Before an evacuation, we should find these bottlenecks and try to improve their passing capabilities to shorten the evacuation time. From the above 6 experiments we found that road 1 and road 2 are the bottlenecks when evacuation demand is large enough. Thus we conducted other several experiments by changing the capabilities of road 1 and road 2 proportionally. The results are shown in TABLE.II.

TABLE.II. EVACUATION TIME UNDER DIFFERENT ROAD SEGMENT CAPACITY

Road 1	60	90	120	150	180	210	240	270	300
Road 2	40	60	80	100	120	140	160	180	200
Time (min)	329	223	175	145	128	119	111	198	234

From deeper analysis we can see that evacuation time increases nearly exponentially when decreasing the capacities of road 1 and road 2 under certain bound (relatively 240 and 160). Intuitionally, the larger the capacities of road 1 and road 2 are, the faster the evacuation is. But it doesn't seem like that when the capacities are too large (when larger than 240 and 160). The minimum evacuation time happens only when the capacity have neither too small nor too large value. That is because too many vehicles go through road 1 and road 2 in a short time and accumulate in the downstream roads, resulting in terrible congestions. This reminds us that in actual evacuation it works little when improvements of road condition are only made at some certain roads. Instead we should try to make a balance among the whole evacuation road network.

IV. CONCLUSIONS

To address the emergency traffic evacuation problems, we build an Emergency Traffic Evacuation Management Platform (ETEMP) for urban areas. The programming work is done in Java and based on Eclipse RCP plugin development technology. In section III of this paper we give an instance to demonstrate how the platform is used in actual management case. The instance at the same time proves the effectiveness of the platform.

In spite of the success, there remains a lot to be improved. The platform we built in this paper is rather a prototype than a full version. For example, we ignore the authority management in architecture design which is critically important especially in cases where security is especially concerned. Secondly, it is only allowed to import existing map data instead of providing an interface to directly create and edit map in the platform. Finally, the decision support module is not intelligent enough to produce practical and useful advices for users. Manual experiences and analysis are needed to make the final decision. We will do further work to get these problems solved.

ACKNOWLEDGMENT

This work was supported in part by National Natural Science Foundation of China projects 71232006, 61233001, 90920305, 61174172, 61104054 and 61203079, 61203166; CAS projects 2F11N01, 2F11N07 and 2F12E05.

REFERENCES

[1] Zhaoming Lu, Peng Lin, Hechao Huang, "The city urgent evacuate system based on GIS", China Public Security, Academy Edition, No. 2, pp. 35-40, August 2005.

[2] Jun Hu, "Data access model based on the MVC Design Pattern", Computer and Information Technology, Vol. 18, No. 5, 2010.

[3] Yi Zhang, "Research of dynamic requirements oriented software modularity structure evaluation", Chongqing: Chongqing University, 2009.

[4] ESRI, "ESRI Shapefile technical description", http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf, 1998-07.

[5] Dong Yang, "The interpretation of Shapefile format and read-writing the data", Journal of Capital Normal University, Natural Science Edition, Vol. 31, No. 2, 2010.

[6] Jintao Wang, Qingchang Meng, "The Eclipse SWT/JFace development practical solution", Beijing: Posts & Telecom press, 2007.

[7] Yun Yuan, "Analysis of Eclipse RCP and application research", Shanghai: Tongji University, 2007.

[8] Gang Chen, "Methods and practices of Eclipse RCP application development", Beijing: Electronic Industry Press, 2007.

[9] Lars Vogel, "Eclipse RCP tutorial", http://www.vogella.com/articles/EclipseRCP/article.html. 2012-02-22.

[10] Jeff McAffer, Jean-Michel Lemieux, Chris Aniszczyk, “Eclipse Rich Client Platform, Second Edition”, Beijing: Tsinghua University Press, 2011.

[11] Ping Zhou, Wenjun Ye, “Application of JfreeChart in statistics and analysis of financial data”, Journal of Chongqing Institute of Technology, Vol. 22, No. 11, 2008.