PROCEEDINGS OF SCIENCE

Design and Implementation of Health Management System Based on iOS Platform

Fei Xu[12]
CAD and Network Technology Research Institute,Dalian University of Technology
No.2 Linggong Road, Ganjingzi District, Dalian City, Liaoning Province,Dalian, 116023,China
E-mail: 761866521@qq.com

Yanping Hu
CAD and Network Technology Research Institute,Dalian University of Technology
No.2 Linggong Road, Ganjingzi District, Dalian City, Liaoning Province,Dalian, 116023,China

Yahui Cheng
CAD and Network Technology Research Institute,Dalian University of Technology
No.2 Linggong Road, Ganjingzi District, Dalian City, Liaoning Province,Dalian, 116023,China

Yan Zhang
CAD and Network Technology Research Institute,Dalian University of Technology
No.2 Linggong Road, Ganjingzi District, Dalian City, Liaoning Province,Dalian, 116023,China

Zhe Sun
CAD and Network Technology Research Institute,Dalian University of Technology
No.2 Linggong Road, Ganjingzi District, Dalian City, Liaoning Province,Dalian, 116023,China

The health management system based on iOS platform was designed and developed to meet the needs of users' personal health management. It can be divided into three function modules, namely, the measurement records management, the diet analysis and the message receiving. The system was also implemented in combination with the latest Swift programming language and the Objective-C APIs. In the realization process, Quartz 2D was used to draw graphs, which can help users to understand their own health trends; with the search display controller, the implemented keyword search function can facilitate the users to find food in the list quickly; and by means of data synchronization technology and JSON parsing technology, the operating efficiency of the system was also improved.

CENet2015
12-13 September 2015
Shanghai, China

[1]Speaker
[2]Fund project: major national science and technology projects(2011ZX04015-021)

© *Copyright owned by the author(s) under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (CC BY-NC-ND 4.0).*

PoS(CENet2015)057

1. Introduction

With the fast growing mobile technology, the number of newly-developed mobile medical software has increased dramatically. This kind of software can not only provide a large amount of clinical information to the healthcare practitioners, but also help improve the users' own abilities of health management. [1-3]. In June 2014, Apple launched a new programming language called "Swift", which is simple, efficient and easy to learn [4]. When we are enjoying Swift's new features, we can also take advantage of Objective-C APIs.

In order to meet the demands of health management and be in accordance with the user-centered principle [5], a health management system was designed, by which the users can record their own health data and develop health life plans effectively; therefore, this application can help users prevent diseases and work as an effective therapy assistant. To help the users understand their own health status better, the measurement records are presented in the form of both lists and graphs. In order to ensure that users can find the food and record in short time, the keyword searching function based on food names is provide; at meanwhile, by leveraging the data synchronization technology, JSON parsing technology and multi-threading technology, the amount of network data transmission has been reduced with the time of page loading shortened.

2. System Design

2.1 Design Pattern of System

In MVC (Model-View-Controller) pattern, the view and the model are separated; the controller is the bridge between the view and the model [6]. The system architecting based on MVC pattern is shown in Fig. 1.

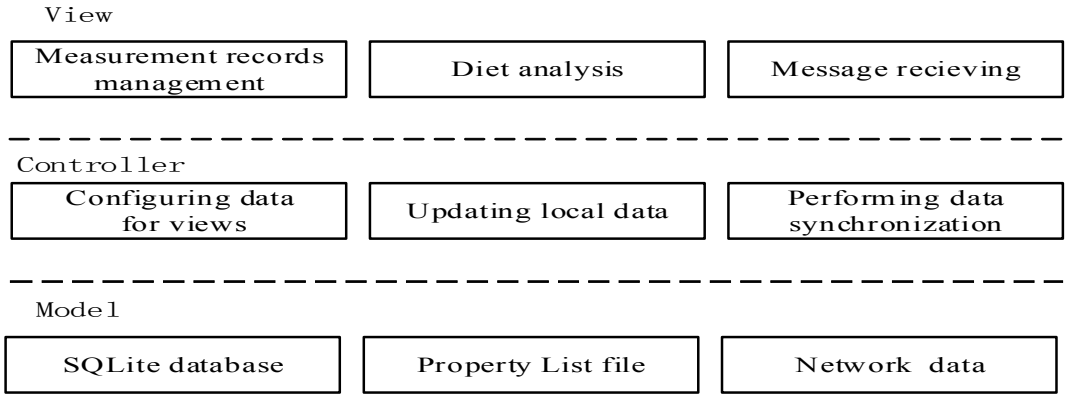

Figure 1: System Architecting based on MVC Pattern

PoS(CENet2015)057

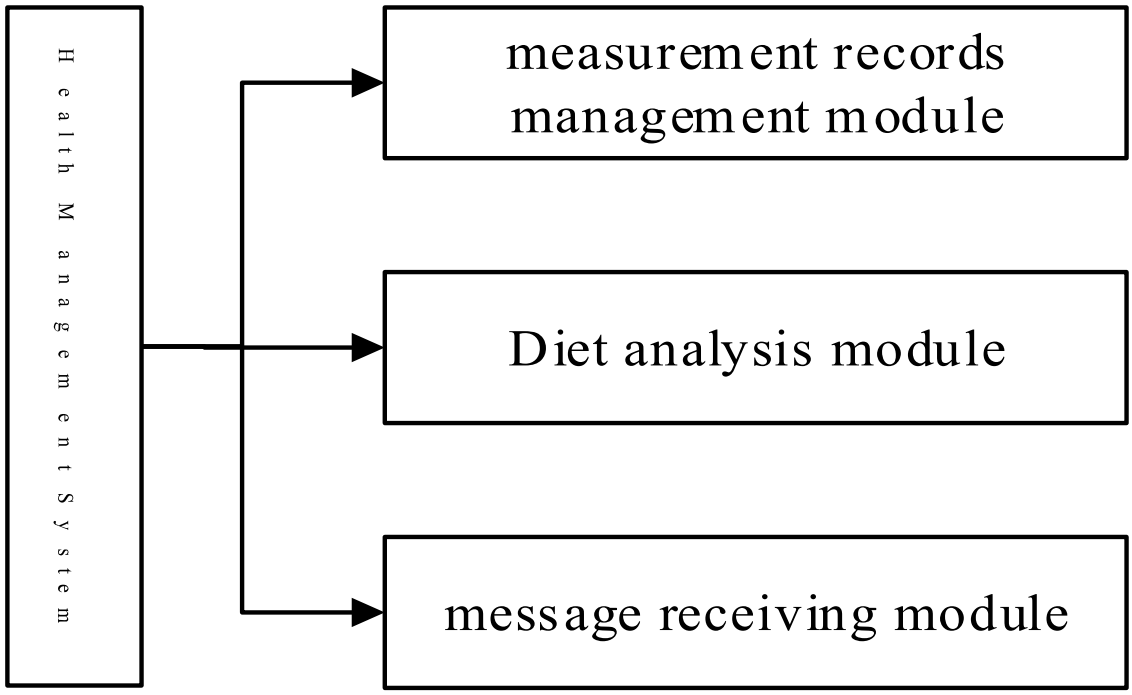

Figure2: Systematic Function Module

(1) View. It is responsible for interaction with users. The recorded data can be converted into the form of lists or graphs and it allows users to add, change and delete the data.

(2) Controller. It is used to configure the data source and obtain changed data from views. After logic verification, it will update local data and synchronize data.

(3) Model. It can be divided into two parts: the local data and the network data, in which, the former is stored in the SQLitle3 database and Property List file, while the latter is in JSON format which needs to be parsed.

2.2 System Functional Module

As shown in Fig. 2, the health management system consists of three functional modules, including the measurement records management, the diet analysis and the message receiving.

(1) Measurement records Management. In this module, four measurement indicators was provided, including blood pressure/pulse, blood lipids, weight/BMI and blood sugar index. The measurement records can be converted into forms of lists or graphs. As to abnormal data, special tips will be shown to the users. No matter the network is connected or not, user can add, delete and change individual measurement records.

(2) Diet analysis. Based on the diet records, analysis of daily intake calories and intake nutrition can be provided, which can assist users to make a reasonable adjustment for diet. Meanwhile, the keyword searching function can facilitate users to find the food and record quickly.

(3) Message receiving. This module includes care tracking, health activities, health appointment and health Information. Care tracking means the health consultants can focus on user's data and give appropriate advices timely; health activities, service appointment, and health information are used to provide recommended wholesome activities, services and health tips.

PoS(CENet2015)057

3. Realization of Key Functions and Critical Technologies

3.1 Measurement Data Graph

Apple offers two graphics libraries: Quartz 2D and OpenGL ES. Between them, Quartz 2D is an advanced and two-dimensional drawing engine. While the measurement graph is two-dimensional as well, Quartz 2D was utilized to complete the drawing of measurement graphs.

3.1.1 Drawing the Coordinate Axis

In the graph, the horizontal axis represents the recording time and the vertical axis represents the range of measurement values. The process of drawing axis can be divided into two steps. The first step is to draw a line segment, which represents the coordinate axe. The second step is to draw calibration labels of the coordinate axes. Taking the process of drawing the X- axis line segment as an example:

(1) UIGraphicsGetCurrentContext () is called to return the current graphics context;

(2) The graphics context is used to save the drawing information (the start point and end point) and the drawing status (color). At the same time, the line segment is drawn in the graphics context;

(3) CGContextStrokePath () is called to paint a line along the current path in views.

The process of drawing the line segment, which represents the Y-axis, is the same.

The scale labels are custom UILable class, their position and size are set by CGRect () function. They are added into the view by addSubview () function. The time range of X- axis is optional. Accordingly, the scale labels are dynamic and putted equidistantly along the X-axis direction. The position is determined by the length of X-axis and the number of labels; the content has a great relationship with the interval scale, which can be calculated by the number of labels and the displayed time range.

The process of setting scale labels of Y- axis is similar to the scale labels of X- axis: they are put equidistantly along the Y-axis direction and the content is determined by both the range of measurement values and the number of labels.

3.1.2 Drawing the Measurement Curve

The process of drawing measurement curve is shown as follows:

(1) Records belonging to the period and selected by the users are filtered out;

(2) The selected records are mapped into points in the graph and positions of the points are calculated as follows:

$$X = \frac{\textit{the recording time} - \textit{the start time}}{\textit{the period of time}} \tag{3.1}$$

$$Y = \frac{\textit{the record value} - \textit{the minimum value}}{\textit{the range of values}} \tag{3.2}$$

(3) The curve is a polyline, which is composed of points in (2). In order to save the drawing information, CGPathCreateMutable () function is leveraged to create a path, and the path was assigned to the path property of CAShapeLayer. In combination with CALayer and CABasicAnimation, an animation effect can be added in the graph. The key code of plotting the curve is as follows:

```
var path = CGPathCreateMutable()
CGPathMoveToPoint (path, nil, axisInset + xAxis [0], self.bounds.height - yAxis [0] - axisInset)
for index in 0..< xAxis.count {  . . .
CGPathAddLineToPoint (path, nil, xValue, yValue) }
var layer = CAShapeLayer()
```


PoS(CENet2015)057

```
layer.path = path
. . .
self.layer.addSublayer(layer)
var animation = CABasicAnimation(keyPath: "strokeEnd")
animation.duration = animationDuration
animation.fromValue = 0
animation.toValue = 1
layer.addAnimation(animation, forKey: "strokeEnd")
```

3.2 Keyword Searching Based on Food Names

When adding diet record, the users have to find the food to be recorded quickly. To meet the needs, keyword searching based on the food names is provided. The keyword can be filled into the search bar by two means: (1) input keywords directly by users; (2) click the history list and the keyword will fill the search bar automatically. According to the keyword in the search bar, the program will filter the data collection, and give the search results quickly.

3.2.1 Table View of the Search History

The last 20 searched keywords will be stored in the local searching history in respect of the users' food and the searched keywords don't have to be synchronized to the server; thus the plist file can be used to save the searched keywords. By using the GCD multithreading technology, the query keyword can be saved to the plist file without reduplication in the query of food. When the search for food interface is loading, the records stored in plist file will be taken out and configured as a data source to a table view (UITableView) below the search bar.

There are two kinds of cells (UITableViewCell) in the table view: one is the custom cell, used to display the food searched records. When the user clicks the cell, the search display controller (UISearchDisplayController) will be activated by setActive () function and the searched keyword in the cell will be filled into the search bar. The other is the original cell, used to support users to clear the search records in the plist file. When the cell is clicked, the plist file will be cleared and the table view of the search history will be refreshed.

3.2.2 Table View of the Search Results

Predicate (NSPredicate) is selected to filter data because it can specify the data filtering method, features the advantage of high efficiency and is convenient to write. The food names containing the keyword can be searched out by categories and the search results are presented in a table view. The key code of filtering data is shown as follows:

```
var pred: NSPredicate = NSPredicate(format: "SELF CONTAINS %@", searchText)
for item in categoryArr  {
var matches:Array  =  self.foodNameDic[item]!
var filteredArray:Array<String> =
matches.filter{ pred.evaluateWithObject($0) }
if(filteredArray.count != 0){
self. filteredFoodDict.updateValue (filteredArray, forKey:item )
self,filteredSection.append(item)}
```

Upon filtering out the qualified data, the searching display controller use its own table view to display the search result, and all the cells in the table view are customized. Because the table view of search results and the table view of history records share the same view controller, the tag values should be used to distinguish the two tables. Of course, the data source methods and the delegate methods in the two tables are different, as shown in Table 1.

PoS(CENet2015)057

Method	Table view of history records	Table view of search results
numberOfSections InTableView:	1	The number of food categoriesin search results
tableView: titleForHeaderInSection:	"History list"	The name of food categories in search results
tableView: umberOfRowsInSection:	the number of history records +1	The number of searchedfood in this category
tableView: cellForRowAtIndexPath:	the history records cell/the delete cell	the search results cell
tableView: didSelectRowAtIndexPath:	jump to search results interface/clear local records in plist files	jump to the food details interface

Table 1 : Processing Methods of Table Views

3.3 Data Synchronization Based on Timestamp

By using the data synchronization technology, the amount of data transmission will be decreased and the speed of program loading will be increased. The primary concern of data synchronization is capturing the change data. The timestamp-based capture method features the advantages of high efficiency, less conflict and easiness of use [7], so the timestamp-based method is adopted.

3.3.1 Design of Synchronization Tables

In order to record data submission time, UpdateTime Field should be added to both the synchronization tables of both server-side database and client-side database. The processing of deleted data is different from that of both the added data and the updated data. In order to identify whether a record is deleted, IsDelete Field is added to the synchronization tables of the server-side database and client-side database; when the network connection status switches from disconnection to connection, the offline records stored in local will be synchronized to the server. In order to identify whether all the records are uploaded to the server-side database, State Field is added to the tables of the client-side database.

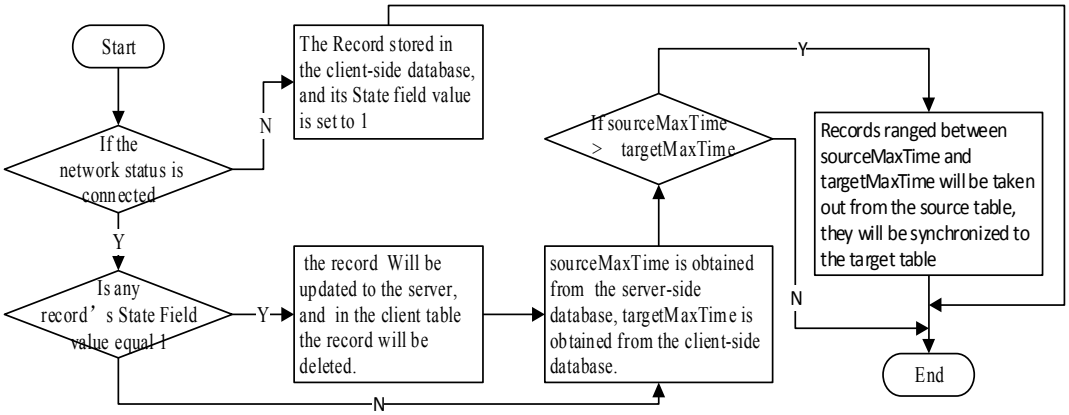

Figure 3: Flowchart of Data Synchronization

3.3.2 Capture of the Change Data

In the timestamp-based capture method, the record's update time is an important basis and recorded in tables as an UpdateTime Field value. In the process of synchronization, the maximum value of UpdateTime Field should be taken out firstly from the source table, i.e. sourceMaxTime, so is the maximum value of UpdateTime Field in the target table, i.e. targetMaxTime; then sourceMaxTime and targetMaxTime will be compared. If sourceMaxTime > targetMaxTime, records ranged between sourceMaxTime and targetMaxTime will be taken

PoS(CENet2015)057

out from the source table, and synchronized to the target table; or, records in the source table is consistent with records in the target table, so no update is needed; besides, there are two special conditions should be taken into consideration:

(1) Deleted Data Synchronization. After the record is deleted by users, its IsDelete Field value will be set to 1. When the record has been deleted from the target table, the record will be still in the source table temporarily and actually deleted from the source table untill the change is totally captured.

(2) Offline Change Data Synchronization. When the network is disconnected, the record changed by the users will be stored in the SQLite database; at the same time, the State Field value will be set to 1, which indicates that the record has not been uploaded to the server database. Once the network connection status switches from disconnection to connection, the offline record will be uploaded to the server; at the same time, the record's update time will be changed. In order to reduce the data conflict, when the offline change data have been successfully updated to the server, all the offline records in the client table will be deleted accordingly. By comparing sourceMaxTime and targetMaxTime, the changed data in server-side database could be obtained and will be synchronized to the client-side database.

3.4 JSON Parsing

Since iOS5.0, Apple provides native support for JSON, i.e. NSJSONSerialization class. Compared to the popular third-party JSON Parsing library, such as TouchJson, SBJson and JSONKit, NSJSONSerialization highlights advantages in terms of usability and parsing efficiency; in this sense, NSJSONSerialization class is selected to implement the parsing of data in JSON format.

By NSJSONSerialization, Objects and arrays in JSON would be converted to the dictionary type (NSDictionary) and array type (NSArray) respectively. Taking the measurement items records in JSON format as an example, the records can be seen as a collection of key-value pairs, named responseDic. There is only one key-value pair in responseDic with the key of "CheckTemplate", and the value is a four-object array. The value could be converted to a variable of NSArray, named templateArr. When traversing templateArr, the elements in the array will be converted to variables of NSDictionary. Finally, the corresponding value can be obtained through keys. The critical code of data parsing is shown as follows:

```
let responseDic = NSJSONSerialization.JSONObjectWithData(responsedata!, options: NSJSONReadingOptions.MutableContainers, error: nil) as NSDictionary
let templateArr = responseDic ["CheckTemplate"] as NSArray
for template in templateArr {
let templateDic = template as NSDictionary
let Check_Template_Name = templateDic["Check_Template_Name"] as String
let Check_Template_Code = templeteDic["Check_Template_Code"] as String ..... }
```

PoS(CENet2015)057

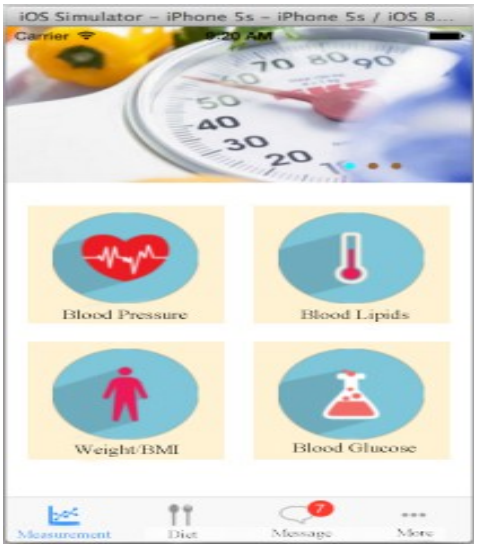

Figure 4: Navigation of Measurement Items

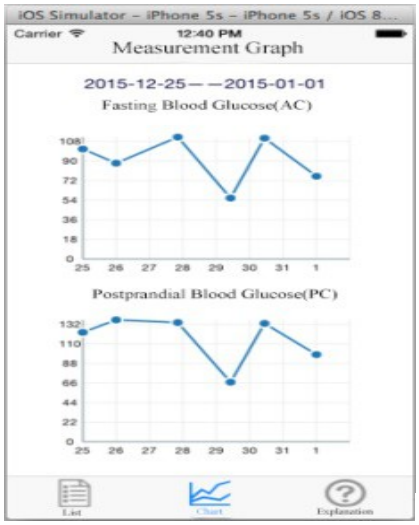

Figure 5: Graphs of Measurement Records

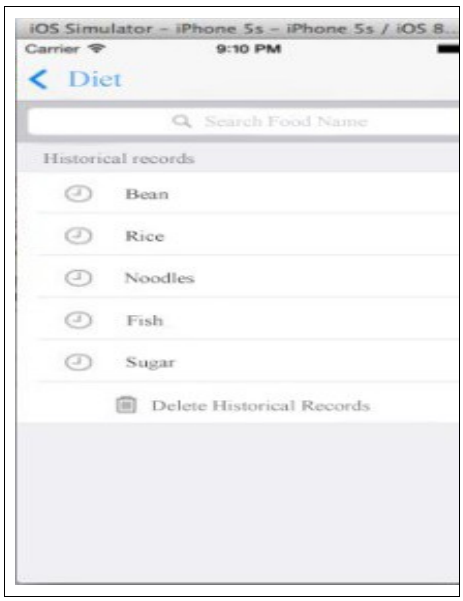

Figure 6: Navigation of Searched Keywords

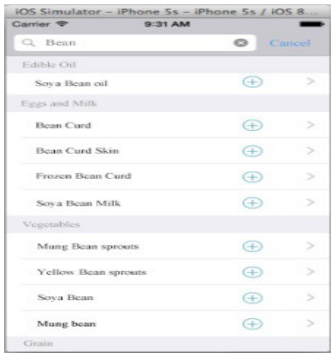

Figure 7: Table View of Search Results

Figure 8: Message Navigation

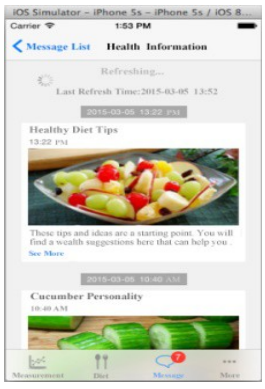

Figure 9: Overview of Health Information

PoS(CENet2015)057

4. Test Results

The test platform was iOS8.0, and the efficiency and convenience of the system were tested. The navigation interface of measurement items is shown in Fig. 4. Upon clicking on the icon, users will enter the sub-page of the corresponding item. As shown in Fig. 5, the graphs of measurement records can help users understand their own health trends better. Fig. 6 and Fig. 7 are the keyword searching interfaces based on food names, the contents of the table view in Fig. 6 are keywords searched by users. The search results can be presented by category, as shown in Fig. 7. The message navigation interface is shown in figure 8. An overview of health information is shown in Fig. 9 and the latest information will be loaded after the "pull down to refresh" behavior.

5. Conclusion

The health management system based on iOS platform was implemented. It can completely support users to carry out personal daily health management. Such mobile development technologies as data synchronization based on timestamp, JSON parsing and GCD, are well integrated into the system. Of course, those technologies can improve the system efficiency. In addition, the system is written in the form of combining Swift language with Objective-C APIs. In the future study and research, the interactive mode could be further improved and the measurement data graph could be presented in a more concise and intuitive way.

References

[1] W. R. Hao, Y. H. Hsu, K. C. Chen, H. C. Li, U. Iqbal, P. A. Nguyen, et al. *LabPush: A pilot study of providing remote clinics with laboratory results via short message service (SMS) in Swaziland, Africa–A qualitative study* [J]. Computer methods and programs in biomedicine.118 (1): 77-83(2015).

[2] S. Doyle-Lindrud, *Mobile Health Technology and the Use of Health-Related Mobile Applications*[J]. Clinical journal of oncology nursing.18 (6): 634-636(2014)

[3] O. El-Gayar, P. Timsina, N. Nawar, W. Eid. *Mobile applications for diabetes self-management: status and potential*[J]. Journal of diabetes science and technology.7 (1): 247-262(2013)

[4] D S. Guan, Z R. Zhao. *Swift Development Guide* [M]. Beijing: Post and Telecom Press.9: pp.226-237(2014). (In Chinese)

[5] K Y. Lin, M H. Tsai, U C. Gatti, J. J. C. Lin, C. H. Lee, & S. C. Kang. A *user-centered information and communication technology (ICT) tool to improve safety inspections* [J]. Automation in Construction.48: 53-63(2014)

[6] Z F. Ren, M S Y. *Overview of the research in model-view-controller pattern* [J]. Application Research of Computers.10: 1-4(2004). (In Chinese)

[7] H W. Zhen. *Research and Design of Embedded Database Synchronization System Based on the Mobile Terminal* [D]. Guangzhou: Guangdong University of Technology (2013).(InChinese)

PoS(CENet2015)057