2004

Morder-Client Food Service

Li Qui

MORDER-CLIENT FOOD SERVICE

A Project

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

---

by

Li Qiu

June 2004

MORDER-CLIENT FOOD SERVICE

A Project

Presented to the

Faculty of

California State University,

San Bernardino

---

by

Li Qiu

June 2004

Approved by:

[redacted] 2/23/04

Dr. Richard Botting, Chair, Computer Science Date

[redacted]

Dr. David A. Turner, Computer Science

[redacted]

Dr. Ernesto Gomez, Computer Science

ABSTRACT

mOrder-Client Food Service can improve service quality and efficiency because it uses Pocket PC, network communication, Wi-Fi and multimedia technologies. It has three parts: Pocket PC, Counter and Kitchen. mOrder-Client Food Service needs some improvements for commercial purpose. If the improved version of mOrder-Client Food Service combines with the Ambol POS, it will become very good commercial software.

ACKNOWLEDGMENTS

Firstly, I want to thank California State University, San Bernardino (CSUSB), Computer Science Department of CSUSB, faculty and staffs of CSUSB. They give me a chance that I can study at CSUSB.

Secondly, I like to thank my advisor, Dr. Botting. He gave many useful suggestions about this project. I'm grateful Dr. Turner and Dr.Gomez's advice during the project.

I'm very thankful to my parents and my brother. Without their help and supporting, I don't have today. I also think all people who give me help when I study in CSUSB.

Finally, I thank MyStore Café. The menu and table layout in the project are based on MyStore Café.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

SOFTWARE REQUIREMENTS SPECIFICATION

1.1 Introduction

The contents of Chapter One present an overview of the project. The contexts of the problem are discussed followed by the purpose, significance of the project, and assumptions. Next, the limitations that apply to the project are reviewed. Finally, definitions of terms are presented.

1.2 Purpose of the Project

mOrder-Client Food Service is the client side of the mOrder food service. mOrder-Client Food Service is a "thin client" basing on a handheld mobile computer. It communicates via wireless (Wi-Fi) with a PC to achieve the purposes of mOrder food service.

1.3 Context of the Problem

In traditional restaurants, waiters/waitresses write customers' orders down and send them to a cook. If the restaurant's business is good, waiters/waitresses will be weighed down with work and maybe make some mistakes when s/he is busy to write orders. Moreover, bad writing always makes a cook do the wrong cooking. It wastes time and money.

1.4 Significance of the Project

In the mOrder-Client Food Service, a waiter/waitresses orders meals via a wireless handheld device. And by reading order items listed on the screen of a personal computer which is in a kitchen, the cook can make the correct dishs. This system can improve service quality and efficiency. The mOrder-Client Food Service is responsible for

- From mOrder-Client, waiters/waitresses can check which table is available in the restaurant.
- mOrder-Client helps waiters/waitresses serving a lot of people in a short time. Waiters/waitresses do not have to go back and forth from the kitchen to customers frequently to send order list and check which table's order is done.
- mOrder-Client avoids waiters/waitresses' bad writing that makes cooks do the wrong meals.

1.5 Assumptions

The following assumptions were made regarding the project:

1. This project requires knowledge of network communication, web server, database, eMbedded

programming, hardware programming, network programming and commercial programming.

2. The waiter/waitress need to know how to use the Pocket PC handheld device.
3. A handheld device is Pocket PC 2002 with Wi-Fi function.
4. The operating systems for the desktop computers are Windows XP Professional with IIS Web Server and Microsoft Access Driver for counter side and kitchen side.
5. Counter side's computer has a database of mOrder-Server Food Service which is developed by Chieh-Chou Chou[1].

1.6 Limitations

During the development of the project, a number of limitations were noted. These limitations are presented here:

1. Because Handheld devices and Wi-Fi are new technologies, there are many problems such as power consumption, the coverage range of radio signal and the interrupt of the environment (especially many kitchen and home equipments' work frequency is same with Wi-Fi), the hardware

information and some operating system's setting of the Pocket PC, such as battery's status, network status, front and background light, etc., are becoming very important information for a real wireless handheld commercial system. Unlike desktop computer, it isn't easy task for system developer of Handheld although they offer some solutions, such as Pocket PC SDK. An applied program which has the function to monitor and control hardware status and settings has commercial value.

2. Only the devices which integrate Wi-Fi chip or which have Wi-Fi adapter's slot, such as CF slot, and these devices have high capacity batteries and the operating system is Pocket PC 2002 or later, suit commercial purpose.

3. The speed of wireless networks and executing programs of handheld devices are much slower than today's wired network and personal computer. So, good architecture of software and programming skill are very important. Low level programming languages are better than high level languages. But it makes the development time of software

becomes very long and the quality of the software difficut to control.

4. Using distributed database technology to solve date access and exchange is a simple and reliable solution. In this way, the server side needs to install a Microsoft SQL Server 2000 database and client side needs to install Microsoft SQL Server CE and ADOCE driver. Although it can simplify the data access and exchange between Counter computer and Pocket PC and improve the software development speed, the whole cost of the system will be very high because of using Microsoft SQL Server 2000 and Microsoft SQL Server CE.

5. Handheld device's (including Pocket PC) hardware and system's setups are easy modified by users and without any privilege limitations. It makes the maintenance of applied systems very difficult.

6. The emulators of handheld devices, including Pocket PC, don't support retrieve and setup hardware and Wi-Fi information. Many system informations also can't be got and set. So, emulators can't be used for development a real

commercial application when it involves hardware and system programming.

7. eMbedded Visual Basic has some defects, such as not supporting user definition class, user definition control, control sets, etc. So, a high skill of programming is needed in complex applied programs.

1.7 Definition of Terms

The following terms are defined as they apply to the project.

Table 1. Definition, Acronyms and Abbreviations

Handheld device	A handheld computer is a computer that can conveniently be stored in a pocket (of sufficient size) and used while you're holding it. Today's handheld computers, which are also called personal digital assistants (PDAs), can be divided into those that accept handwriting as input and those with small keyboards.

Windows CE.NET PocketPC 2002 & Smart Phone 2002	They are based on the Microsoft Windows operating system but are designed for including or embedding in mobile and other space-constrained devices. They are 32-bit multitasking, multithreading operating systems. They support SSL, VPN, 40- and 128 bit encryption, etc.
802.11b(Wi-Fi)	802.11 is a family of specifications for wireless local area networks (WLANs) developed by a working group of the Institute of Electrical and Electronics Engineers (IEEE). There are currently four specifications in the family: 802.11, 802.11a (WiFi5), 802.11b (WiFi), and 802.11g. All four use the Ethernet protocol and CSMA/CA (carrier sense multiple access with collision avoidance) for path sharing. The 802.11b standard - often called Wi-Fi - is backward compatible with 802.11. The modulation used in 802.11 has historically been phase-shift keying (PSK). The modulation method selected for 802.11b

	is known as complementary code keying (CCK), which allows higher data speeds and is less susceptible to multipath-propagation interference.
GUI	Graphical User Interface. The graphical representation of physical or pseudo-physical objects (such as buttons, trees, and lists) that allow the user to direct the flow of the program through the use of a mouse or other pointing device.
XML	XML (Extensible Markup Language) is a flexible way to create common information formats and share both the format and the data on the World Wide Web, intranets, and elsewhere.
.NET	Microsoft® .net is the Microsoft XML Web services platform. XML Web services allow applications to communicate and share data over the Internet, regardless of operating system, device, or programming language.
eMbedded Visual	It is an integrated development environment

Tools	(IDE) from Microsoft in which a programmer uses a graphical user interface (GUI) to choose and modify preselected sections of code written in the choice programming language.
Embedded systems programming	Embedded systems programming is the development of programs intended to be part of a larger operating system or, in a somewhat different usage, to be incorporated on a microprocessor that can then be included as part of a variety of hardware devices.
DHCP	The Dynamic Host Configuration Protocol (DHCP) is an Internet protocol for automating the configuration of computers that use TCP/IP. DHCP can be used to automatically assign IP addresses, to deliver TCP/IP stack configuration parameters such as the subnet mask and default router, and to provide other configuration information such as the addresses for printer, time and news

	servers.
SDK	Software Development Kit
IIS	Internet Information Server
POS	Point of Sales.
Ambol POS	Ambol POS is commercial software which is designed and programmed by me for dine-in restaurants, fast food restaurants, kiosks or resellers. Its platform is Windows XP/all-in-one POS machine with touch screen, pole display, magnetic card reader, thermal receipt printer and cash drawer. It supports up to 3 languages at the same time. The database is created automatically when it is run at the first time. It can run in one POS machine situation or in many POS machines with network environment. It also has voice prompting function and blocks other operating system's functions. The software has very easy installation, use and maintenance.

1.8 Organization of the Thesis

The thesis portion of the project was divided into six chapters. Chapter One provides software requirements specification, an introduction to the context of the problem, purpose of the project, significance of the project, limitations, and definitions of terms. Chapter Two consists of the software design. Chapter Three documents the steps used in testing the project. Chapter Four presents the maintenance required from the project. Chapter Five presents the users manual from the project. Chapter Six presents conclusions drawn from the development of the project. The Appendices containing the project follows Chapter Six. Finally, the references for the project are presented.

CHAPTER TWO

SOFTWARE DESIGN

2.1 Introduction

Chapter Two consists of a discussion of the software design. First in this chapter is the preliminary design. Then I'll give the architecture and detailed design. Finally there is the summary of software design.

2.2 Preliminary Design

At the beginning, I want to utilize the distributed database of Microsoft SQL Server and Microsoft SQL CE to achieve the exchange of the data. The good point of this way is that it can make the programming Pocket PC becomes very easy—just like access a local database on PC. But the whole cost of the system will be very high (Refer 1.6 in Chapter One). I also want to use C as the major programming language of the project. Although it is the best language to get hardware and system information, it needs more time on coding and the quality of the program isn't easy controlled. Finally, I decided to use Visual Basic as the main programming language in this project.

2.3 Architecture Design

mOrder-Client Food Service uses Wi-Fi technology to communicates the information between PC and Pocket PC, and shows the information on Pocket PC at GUI style. There are three parts: Pocket PC part, Counter part and Kitchen part. The Pocket PC part's tasks are retrieve table layout, menu, etc., from counter, show table layout, order meal, show ordered items' status and check out. The counter side's taskes retrieve, update or insert data according to Pocket PC's request and send the results to Pocket PC via IIS. The kitchen part is optional part. It can update ordered items' status and send item's name and table number to Pocket PC when the ordered item is ready. See Figure 1.

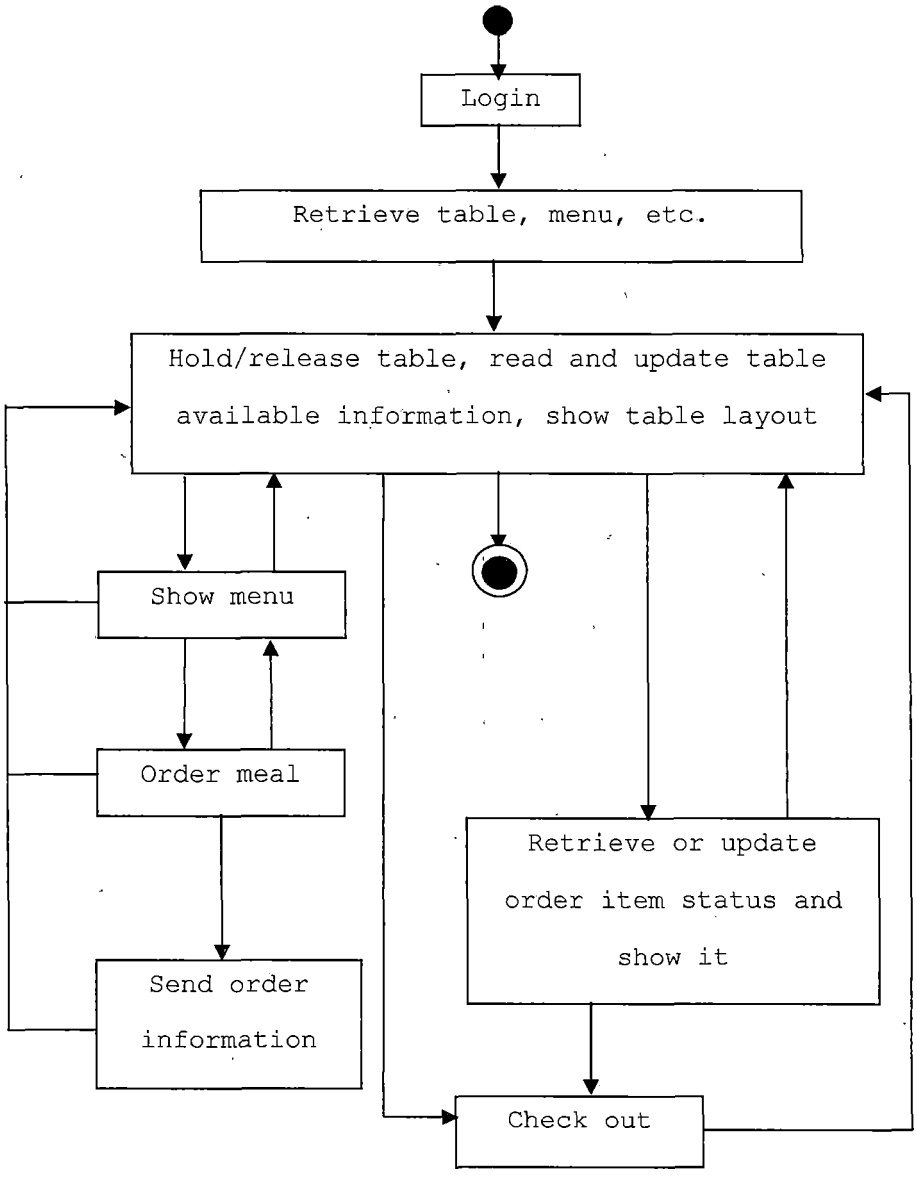


Figure 1. Pocket PC Program

Because eMbedded Visual Basic doesn't support user definition data types, there isn't any class diagram. The use case diagram is

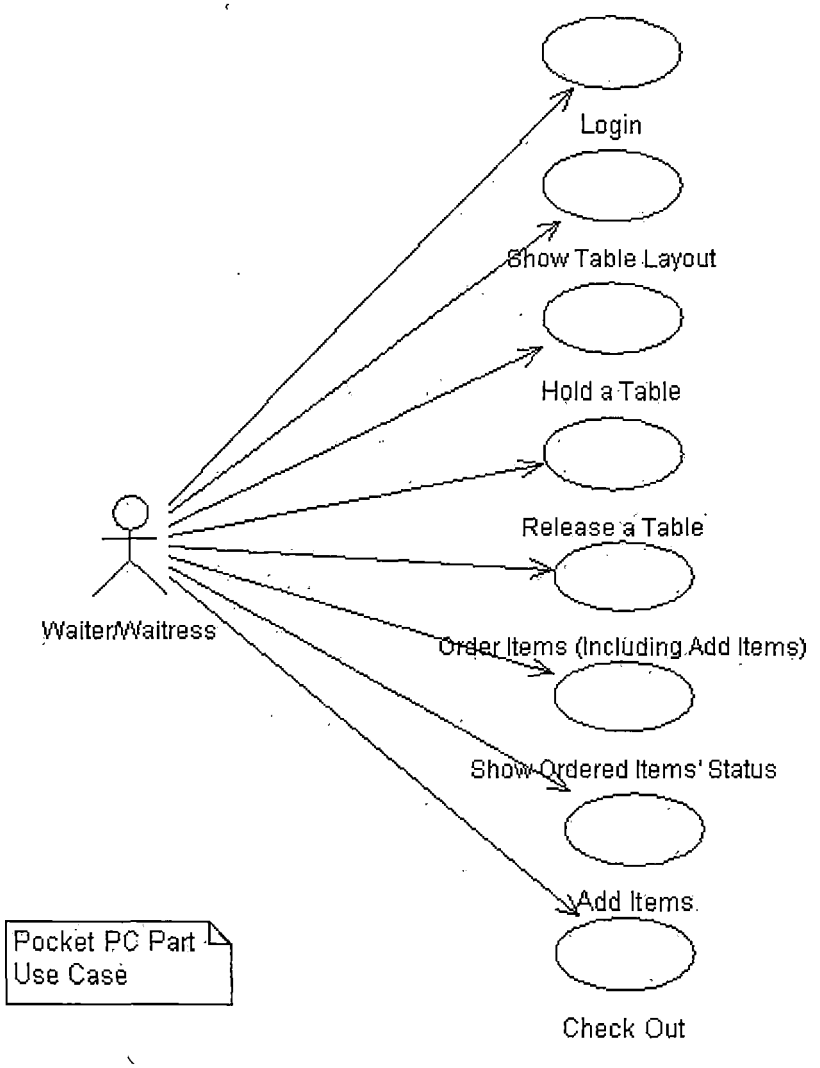


Figure 2. Pocket PC Part Use Case Diagram

The Kitchen Side's structure is

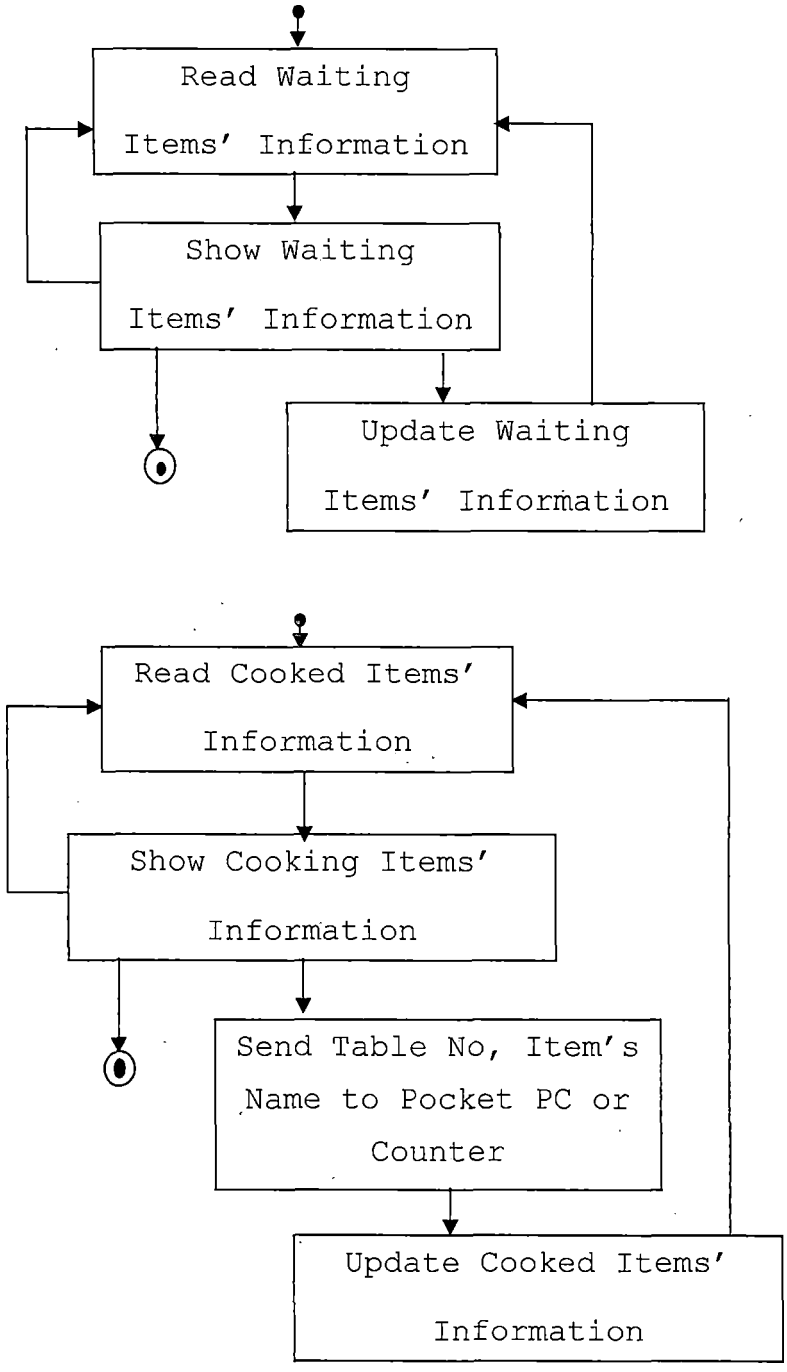


Figure 3. Kitchen Part Program

The kitchen part's use case diagram is

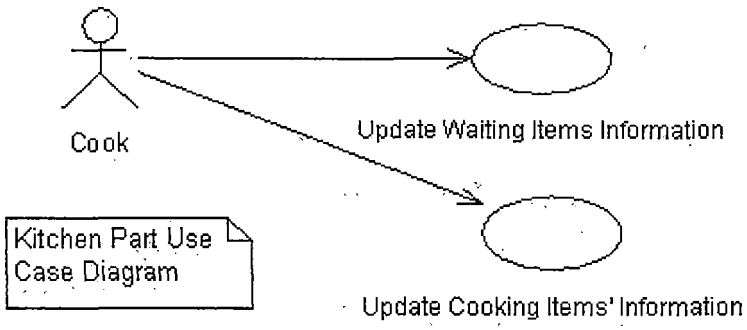


Figure 4. Kitchen Part Use Case Diagram

2.4 Detail Design

2.4.1 Pocket PC Part

The mainly section of this part is that the Pocket PC's program triggers requests to send or get the information by Socket. Then the program shows the information which retrieve from counter PC as GUI style on its screen. A waiter/waitress just tip the wanted control on the screen when it is needed (Please refer 2.5, 3.1, 4.1, etc., in Chapter FIVE). The main programming language in this part is eMbedded Visual Basic.

The following codes illustrate the main aspects of the network communication.

```
Private Sub SendRequest

 Socket.Open
 Socket.ConnectionRequest
 strRequest="request content"
 Socket.SendData strRequest
End Sub

Private Sub DataArrival
        Socket.GetData strHTML
        Treat strHTML
        Socket.Close
End Sub

Private Sub ConnectionRequest
        if Socket.Status <> CLOSED then
                Socket.Close
        End
        Socket.Accept
End Sub

Private Sub Close
        Socket.Close
        Socket.Listen
End Sub

Private Sub Error
        Show error information
        Socket.Close
        Socket.Listen
End Sub

Private Sub SendComplete
        Socket.Close
End Sub
```

Figure 5. Pocket PC Socket Code

Because the eMbedded Visual Basic doesn't support user definition class, user definition control, control sets, dynamic memory allocation, etc., I have to setup many property procedures to deal these problems. The following source code is a property procedure of item's control:

```
Private Sub ItemControl(anItem As Label, index As Integer)

        If mOrderForm.Caption = MenuTableNumberString Then
                        NumPanelTitleLabel.Caption = ItemName(Index, Tabstrip2Index, Tabstrip1Index)
        ElseIf mOrderForm.Caption = "Order List" Then
                        NumPanelTitleLabel.Caption = anItem.Caption
                        ShowStatusFrame True
        ElseIf mOrderForm.Caption = "Order Item Status"   Then
                If anItem.ForeColor = itemReadyColor = itemBlinkColor Then
                anItem.ForeColor = itemOfferColor
                        str1 = OrderItemSeriesNumber(itemIndex)
                        str2 = orderItemOffered
                        UpdateItemStatus
                End If
        End If
End Sub
```

Figure 6. Property Procedure

In fact, there isn't any essential distinction between wired and wireless network communication at application development level. The main difficulty is how we can get and set the device or operating system status from eMbedded visual basic. I already mention previous paragraph that eMbedded Visual Basic has some defects, especially at this area. So we can't get the information directly by using eMbedded Visual Basic. Fortunately, eMbedded Visual C can do this. For example, we just need to call C program in eMbedded Visual Basic as this way to get Pocket PC battery status:

```
Public Declare Function GetSystemPowerStatusEx Lib "Coredll" (ByVal PowerStatus _ As Long, ByVal Update As Long) As
Long
```

Figure 7. Call C Function

Here, the variable PowerStatus is a long variable. It delegates the structure, SYSTEM_POWER_STATUS_EX2.

```
Typedef struct _SYSTEM_POWER_STATUS_EX2 {
        BYTE ACLineStatus;
        BYTE BatteryFlat;
        BYTE BatteryLifePercent;
        BYTE Reserved1;
        DWORD BatteryLifeTime;
        DWORD BatteryFullLifeTime;
        BYTE Reserved2;
        BYTE BackupBatteryFlag;
        BYTE BackupBatteryLifePercent;
        BYTE Reserved3;
        DWORD BackupBatteryLifeTime;
        DWORD BackupBatteryFullLifeTime;
        WORD BatteryVoltage;
        DWORD BatteryCurrent;
        DWORD BatteryAverageCurrent;
        DWORD BatteryAverageInterval;
        DWORD BatterymAHourConsumed;
        DWORD BatteryTemperature;
        DWORD BackupBatteryVoltage;
        BYTE BatteryChemistry;
} SYSTEM_POWER_STATUS_EX2;
```

Figure 8. Pocket PC Battery Status Structure

2.4.2 Counter Part

At the counter side, IIS Web Server gets HTTP requests from a Pocket PC and retrieves, updates or inserts the data of a waiter/waitress, table layout, table available, menu, order information, etc., via ASP Script. For Example, when the Pocket PC requests menu information, the ASP Script will retrieve the menu and send it to the Pocket PC:

```
<% @Language = "VBScript" %>

<%
'-----------------------------------------------------------------------
' File:       ReadItemInfo.asp
' Purpose:    Demonstrate retrieving a recordset
' Version:    1.5
' Author:     Li Qiu
' Date:                 May 22, 2002
'-----------------------------------------------------------------------
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT ItemNumber, ItemName, Price, ITEM.SubcategoryNumber " &_
            "FROM ITEM, SUBFOODCATEGORY " &_
            "WHERE ITEM.SubcategoryNumber=SUBFOODCATEGORY.SubcategoryNumber
                        AND CategoryNumber=Request.QueryString("Parm1") & " "&_
                        "ORDER BY ITEM.SubcategoryNumber, ItemNumber"
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open    "Provider=Microsoft.Jet.OLEDB.4.0;DataSource=c:\morder\mOrderDatabase.mdb"
rs.Open sSQLQuery, oConn
Do Until rs.EOF
        Response.Write(rs("ItemNumber") & ";"_
                        & rs("ItemName") & ";" _
                        & rs("Price") & ";"_
                        & rs("SubcategoryNumber") & ";"_
                        )
        rs.MoveNext
Loop
%>
```

Figure 9. Read Menu ASP Script

2.4.3 Kitchen Part

At the kitchen side, the program shows ordered items in two columns. One is waiting list and another is cooking list. They show ordered items' name, quantity, special demand, time and table number. When a cook want to cook an item in the waiting list s/he clicks it, its cooking status will become as cooking and it will jump from the waiting list into the cooking list. After the cook has cooked the dish and clicks the item in the cooking column, the PC will send the table number, dish's name to the Pocket PC which ordered it via Socket, and the item's status will be

updated as ready. When the Pocket PC gets the information, it shows the information on its screen and rings the waiter/waitress. This part's programming language is Visual Basic.

```
Call Winsock1.Connect
strMessage = bItemName(bIndex) + "," + bTableNumber(bIndex)
Call CounterSocket.Connect
scMessage = speakString

Private Sub Winsock1_Close()
  Call Winsock1.Close
End Sub

Private Sub Winsock1_Connect()
  Call Winsock1.SendData(strMessage)
End Sub

Private Sub Winsock1_Error(ByVal Number As Integer, Description As String, ByVal Scode As Long, ByVal Source As
String, ByVal HelpFile As String, ByVal HelpContext As Long, CancelDisplay As Boolean)
  'MsgBox Description
  Winsock1.Close
End Sub

Private Sub Winsock1_SendComplete()
  Call Winsock1.Close
End Sub
```

Figure 10. Kitchen Part Socket Program

2.5 Summary

The software design of the project was presented in this chapter. There are three parts in the project: Pocket PC, counter and kitchen. The kernel of Pocket PC part is network communication. Using ASP Scripts accessing database and exchanging the information via IIS is the main point of counter side. At kitchen part, triggering socket to send the information to Pocket PC is the core.

CHAPTER THREE

SOFTWARE QUALITY ASSURANCE

3.1 Introduction

Chapter Three documents the software quality assurance. Because mOrder-Client Food Service has three scenarios, and it will integrate with mOrder-Server, there are unit test plan, integration test plan and system test plan.

3.2 Unit Test Plan

According to the project's functions, the counter side must be tested first. Then it is Pocket PC part. Kitchen part test is last part.

1. Counter Part Test Plan. This part's test is done on the counter PC by using internet explorer to call ASP script files via IIS

Table 2. Counter Part Test Plan

Test No	Test Content	Wanted Result	Test Result	Notes
1	Checkout.asp	Update Total, Payment method, End date, end time into the database with the given order number into table Orders	The script can update the wanted data into the given order record	
2	Order.asp	Insert TableNumber,	The test result is	

		BeginningDate, BeginningTime, WaiterNumber, SpecialDemand, AddTimes, PocketPCIPAddress into table Orders and return the order number	the same with the wanted result	
3	OrderItem.asp	Insert OrderNumber, ItemNumber, Quantity, SpecialDemand, AddMoney,Status, AddTimes, Promotion, AddDate, AddTime into table OrderItems	The program can insert all data into OrderItems	
4	ReadAllTableInfo.asp	Read TableNumber, AvailableStatus, SeatNumber, XCoordinate, YCoordinate, WaiterNumber,ReserveTable, Width, Length from table Tables	The program can read all wanted data	
5	ReadFoodCategoryInfo.asp	Read all CategoryName from table Category	The program can read all wanted data	
6	ReadItemInfo.asp	Read all ItemNumber, ItemName, Price, SubcateogryNumber with the given category from table Items	The program can read all wanted data	
7	ReadOrderInfo.asp	Read TableNumber, Total, PayWay, BeginningTime, EndTime, WaiterNumber,SpecialDemand, AddTimes from table Orders	The program can read all wanted data	
8	ReadOrderItemDetailInfo.asp	Read SeriesNumber, ItemNumber as ItemNo, ItemName, Price, Quantity,	The program can read all wanted data	

		SpecialDemand, ddMoney, Status, AddTimes, Promotion from table OrderItems		
9	ReadOrderItemNumber.asp	Read the order number from table Orders	The program can read all wanted data	
10	ReadStoreInfo.asp	Read StoreName, PhoneNumber, Address, City, State, ZipCode, TaxRate from table Store	The program can read all wanted data	
11	ReadSubcategoryInfo.asp	Read selected category's all SubcategoryName, CategoryNumber from table Subcategory	The program can read all wanted data	
12	ReadUncheckoutOrder.asp	Read selected and unchecked out order's OrderNumber,TableNumber from table Orders	The program can read all wanted data	
13	ReadWaiterHallInfo.asp	Check the given waiter/waitress information from table Waiter	The program can read all wanted data	
14	UpdateAddTimes.asp	Update AddTimes of Orders Information	The result is the same with the wanted	
15	UpdateOrderItemStatus.asp	Update OrderItem's status according to SerialNumber in table OrderItems	The result is the same with the wanted	
16	UpdateTableAvailableStatus.asp	Update the given table's available status in table Tables	The result is the same with the wanted	

2. Pocket PC Part Test Plan. There are many contents needed to be tested in this part. I just mention some of them.

Table 3. Pocket PC Part Test Plan

Test No	Test Content	Wanted Result	Test Result	Notes
1	Check the executable file in Pocket PC	The mOrder icon can be found in Start menu	The result is the same with the wanted	
2	The program is executable	It can show login form	It shows login form	
3	Battery Status	It can show battery's status. When the Pocket PC is in its cradle, it shows Charging. Otherwise shows percentage of battery and the percentage must be down with time elapse.	The result is the same with the wanted	
4	NonPocke PC's Software Input Keyboard's function	There are relevant functions when tap the keys	The result is the same with the wanted	Refer chapter 5 (2.4)
5	Login test	Input user name and password to check the the login function	The result is the same with the wanted	Refer chapter 5 (2.3)-(2.10)
6	Table Layout	The table's location, table number, seat number, available status. Refesh function	The result is the same with the wanted	Refer chapter 5 (2.11)
7	Hold a table	Hold any available table	The result is the same with the wanted	Refer chapter 5 (3.1)
8	Add Items or	For a nonheld table/ held table,	The result is	Refer

	order items	we can order/add items for it	the same with the wanted	chapter 5 (3.2)
9	Cancel hold	Tap nonorder hold table to cancel hold it	The result is the same with the wanted	Refer chapter 5 (3.3)
10	Check out	Tap order hold table to check out	The result is the same with the wanted	Refer chapter 5 (3.4) 5
11	Order Item Status	Tap order hold table to check ordered item status	The result is the same with the wanted	Refer chapter 5 (3.6)
12	Order/Add Item in menu	Select category, select subcategory, show menu, add or order items, modify quantity, cancel a item, input special demand	The result is the same with the wanted	Refer chapter 5 from(4.1) to (4.6)
13	Show, modify,delete chosen items, add new items and send the order	Show order list and modify quantity and special demand, delete, add more items and send the ordered information to counter PC	The result is the same with the wanted	Refer chapter 5 from(4.8) to (4.14)
14	Check and update order items' status	Show different ordered items' status and the status can be update when a dish is servced and the information can be refresh and can return table layout form	The result is the same with the wanted	Refer chapter 5.6
15	Exit the program	In any step, you can exit the program by tap ok button, and the program will pop a confirm form	The result is the same with the wanted	Refer chapter 5.7
16	Alarm	When a dish is ready and is ordered from a Pocket PC, it will show the dish's name, table number	The result is the same with the wanted	Refer chapter 5.8

		and it will ring		
17	Network function	When the counter PC or wireless network has problem, the program will pop an alerm message	The result is the same with the wanted	Refer chapter 5 Troublesh ooting 1
18	Form titles	In every step, the title in that step are different	The result is the same with the wanted	Refer chapter 5
19	Autohide Pocket PC's SIP	In any step, tap the Pocket PC's SIP, it will be hide by the program automatically	The result is the same with the wanted	Refer chapter 5
20	Buttom's menu and their function	In every form, there are different Buttom's menu. For example in chapter 5 (3.3), there is Refesh menu. Check every menu's function	The result is the same with the wanted	Refer chapter 5
21	Pop out frames	There are many pop-out frames. For example in chapter 5 (2.4), when users tap login, it pop-out a SIP	The result is the same with the wanted	Refer chapter 5
22	Check category, subcategory, menu	Check every category, subcategory, menu's information	The result is the same with the wanted	Refer chapter 5
23	Check order information	Check every order items information, including name, quantity, special demand	The result is the same with the wanted	Refer chapter 5
24	Check ordered items status	Check every ordered items' status	The result is the same with the wanted	Refer chapter 5
25	Check scroll bar, left, right arrow	Check scroll bar in menu, order list, ordered items' status form	The result is the same with the wanted	Refer chapter 5
26	Check lock hold table's function	When a waiter hold a table, other waiter can't hold or access the table	The result is the same with the wanted	Refer chapter 5 (3.1)

3. Kitchen Part Test Plan. The kitchen program is very simple. There are three aspects: waiting list, cooking list and voice function.

Table 4. Kitchen Part Test Plan

Test No	Test Content	Wanted Result	Test Result	Notes
1	Check waiting list	When a waiter orders items, they must be showed in waiting list group by the table. There are item's name, quantity, special demand, table number, ordered time. The different tables' orders have different color.	The result is the same with the wanted	Refer chapter 5
2	Waiting list	When a cook begin to cook a dish, he click it in waiting list they must be transfer from the waiting list to cooking list. All dishes in the cooking list group by the table. There are item's name, quantity, special demand, table number, ordered time. The different tables' orders have different color	The result is the same with the wanted	Refer chapter 5
3	Voice function	When there are new order items send from Pocket PC or couter, it must accounce "There are xx new order items"	The result is the same with the wanted	

3.3 Integration Test Plan

The integration test mainly focuses on how every part can cooperate smoothly with each other.

Table 5. Integration Test Plan

Test No	Test Content	Wanted Result	Test Result	Notes
1	Pocket PC with counter PC	When a Pocket PC holds/release a table, sends order information, check out or update ordered items' information, the data in counter PC's database must update. Vice versa	The result is the same with the wanted	
2	Pocket PC with Kitchen Side	When Pocket PC send order information, kitchen side's PC must show it in the waiting list. If kitchen side's PC changes the ordered items' status, the Pocket PC can know. Especially when a dish is ready, it will get the information from kitchen PC and ring	The result is the same with the wanted	
3	Counter Side with Kitchen Side	When kitchen side update ordered items' status, the data in the counter side's database must be update. Vice versa	The result is the same with the wanted	

3.4 System Test Plan

System test will test that the two sections of mOrder Food Service, mOrder-Server[1] and mOrder-Client, can work together seamless and don't interrupt each other.

Table 6. System Test Plan

Test No	Test Content	Wanted Result	Test Result	Notes
1	More than 1 Pocket PC	When there are 2 or more Pocket PCs, every Pocket PC works independently, and can't interrupt each other. The kitchen just sends the ready dish's information to the Pocket which ordered it. A table is servced by a Pocket PC, the others can't access it if they aren't the same account.	The result is the same with the wanted	
2	mOrder-Server Food Service with mOrder-Client Food Service.	The mOrder-Server has the same fuctions and role with mOrder-Client Pocket PC part in table service and can be switch with each other except it has management functions. If counter PC order a dish, when it is done, kitchen PC will send the information to the counter PC and the counter PC will say the information via voice. The Pocket PC's hold /release a table, order items, update items' status can be saw on the counter PC. Vice versa	The result is the same with the wanted	
3	Kitchen side PC collapses or doesn't install	If there isn't kitchen side's PC, the whole system can work, however, some functions will lose.	The result is the same with the wanted	

4	Pocket PC collapses or doesn't install	If there isn't Pocket PC, the whole system can work without any problems.	The result is the same with the wanted	

3.5 Summary

Because the project has three parts, multi-user situation and it will combine with mOrder-Server Food Service as a whole system, the test plan including unit test plan, integration test plan and system test plan. The unit test carefully tests every part's design function. The functions of every part can work correctly when they are combined is main purpose in the integration test. In the system test plan, more than one Pocket PC, Kitchen side PC collapses or doesn't install and Pocket PC collapses or doesn't install are the kernel.

CHAPTER FOUR

MAINTENANCE

4.1 Introduction

Unlike many commercial softwares, such as Ambol POS, just to put the release CD into the CD-ROM driver of a computer, the installation programs can be run automatically and hardware and operating system's setup will be done by Ambol POS software when these programs are run at first time, I haven't created installation packages for the three parts. And there aren't many hardware and operating system controls and setups in these applied programs. So, the maintenance of this project is complex. In the following sections, I'll describe how to setup the three parts in turn.

4.2 Pocket PC Part

There are three Files:

Form1.bef

Project1.ebp

Project1.vbw

After installing Microsoft eMbedded Visual Tools 3.0, Microsoft ActiveSync 3.0 or latter version, putting a Pocket PC handheld computer in its cradle, you can put these files in any fold, click Project1.ebp icon to compile

it and send it into the handheld. Please refer the manual of Microsoft eMbedded Visual Tools 3.0 and Microsoft ActiveSync 3.0 to know the more detail information about how to install and use the development tools. The maintenance of this part is described in Chapter FIVE Troubleshooting. The detail usage and maintenance information of Pocket PC can be found in each Pocket PC's manual.

4.3 Counter Part

Copy mOrder folder to C: Drive. This holds the database file, mOrderDatabase.mdb.

Copy ASP File fold to C Driver. There are 16 ASP Script files:

Checkout.asp

Order.asp

OrderItem.asp

ReadAllTableInfo.asp

ReadFoodCategory.asp

ReadHallInfo.asp

ReadItemInfo.asp

ReadOrderInfo.asp

ReadOrderItemDetailInfo.asp

ReadOrderItemNumber.asp

ReadStoreInfo.asp

ReadSubfoodCategory.asp

ReadUncheckoutOrder.asp

ReadWaiterHallInfo.asp

ReadWaiterInfo.asp

UpdateAddTimes.asp

UpdateOrderItemStatus.asp

UpdateTableAvailableStatus.asp

Setup IIS and make the virtual directory alias as mOrder, the web site content directory as C:\ASP File and access permissions as "Execute". More detail information about how to setup IIS please refer Windows XP manual. Setup this computer's name as Counter and its IP address is 192.168.0.10 and reboot it. It is better to set the resolution of screen at 1024x768. The time format is set as HH:mm:ss; data format is set as MM/dd/yyyy. Share the fold mOrder. Set the XP support Chinese. Then install programs:

tv_enua.exe

peedy.exe

spchapi.exe

They are text-to-speech engine, peedy character engine and speech API file.

4.4 Kitchen Part

Install Microsoft Visual Studio 6.0 and its Service Package 5.0, set the computer name as Kitchen, and the IP address as 192.168.0.11. The time format is set as HH:mm:ss; data format is set as MM/dd/yyyy. The resolution of screen is 800x600. Set the XP support Chinese. Reboot it and install programs:

tv_enua.exe

spchapi.exe

peedy.exe

They are text-to-speech engine, peedy character engine and speech API file. Please refer Microsoft Visual Studio 6.0 Manual if you don't know how to install and use it. Then copy the following source files into a fold named as Kitchen Side:

Kitchen.vbp

Kitchen.vbw

KitchenSide.frm

And compile it.

4.5 Summary

To maintain this project, you need to install many development tools, and there are many hardware and software setups. If for commercial purpose, it is better that all

setups are done by applied programs. Most of them will touch hardware control, operating system and network programming, user management, privilege control, software protection, etc., making release CD, installation guide, administrator guide, etc.(Such as Ambol POS)

CHAPTER FIVE

USERS MANUAL

5.1 Introduction

This chapter will give Pocket PC part and kitchen part user manual. Because counter part task is done automatically by IIS, there isn't any user manual in this part.

5.2 Pocket PC Part Users Manual

1. Turn on your Pocket PC

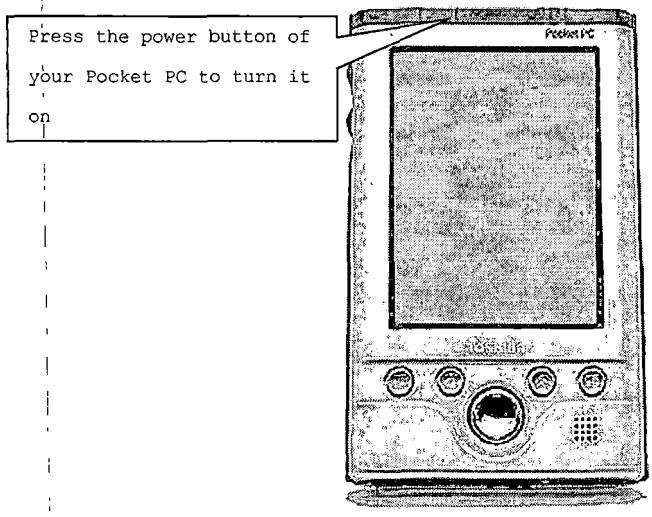


Figure 11. Turn on Pocket PC

2. Login mOrder system

(2.1) Use a stylus to tap  icon on your Pocket PC's screen to pop down a menu

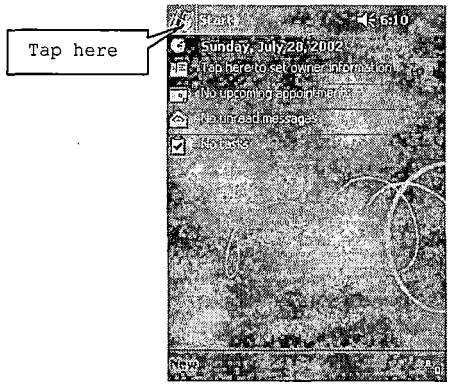


Figure 12. Pop down the Start Menu

(2.2) Tap 

to execute mOrder Pocket PC side's program

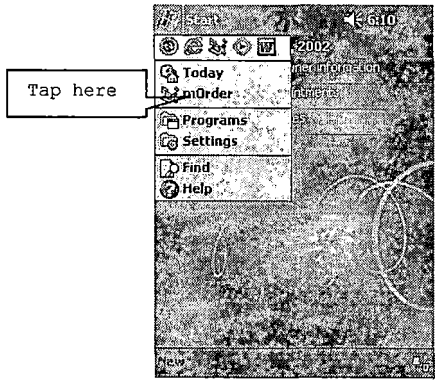


Figure 13. Execute mOrder Program

(2.3) Tap the white space next to the Login label to input your username

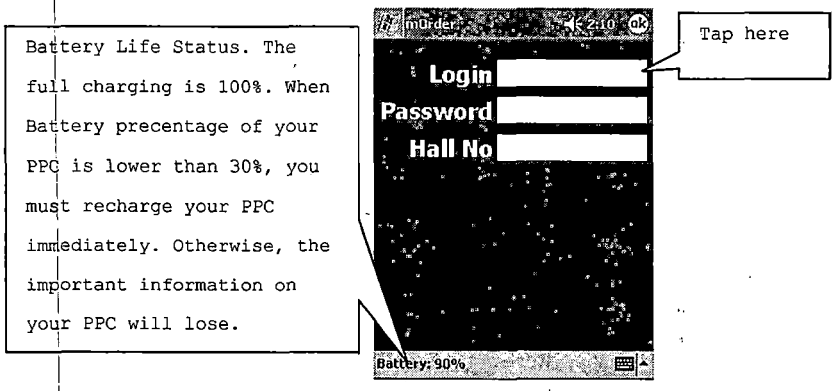


Figure 14. Begin Login

(2.4) After you tap the white space next to the Login label, a sample SIK (soft input keyboard) will be displayed on your Pocket PC's screen

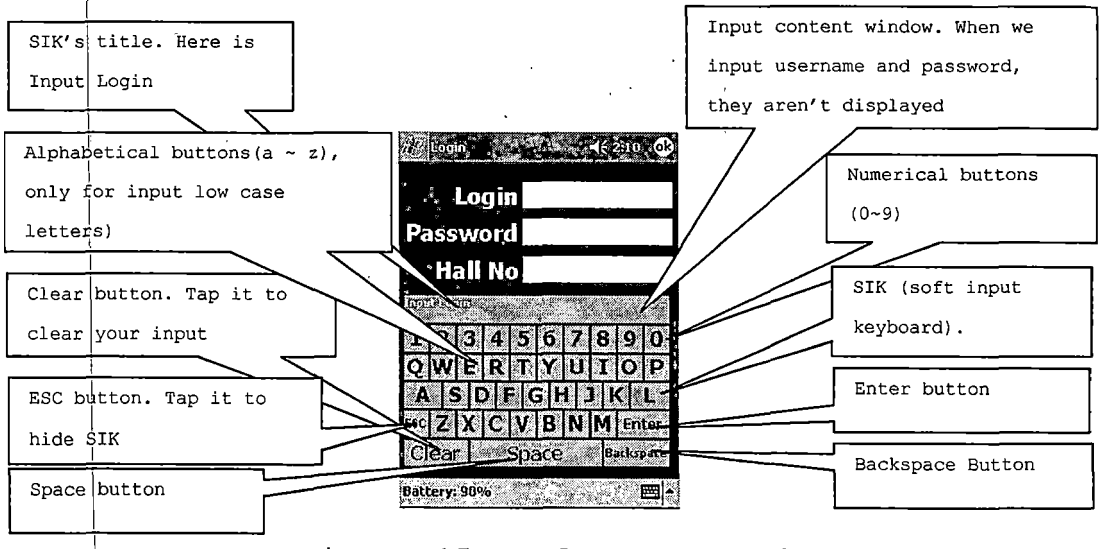


Figure 15. Soft Input Keyboard

(2.7) And input your password and tap Enter button to login mOrder system if you don't want input hall number

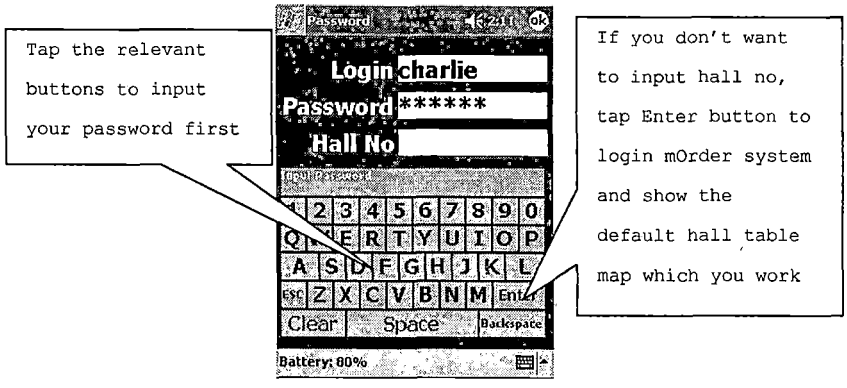


Figure 18. Input Password

(2.8) If you want to input Hall Number

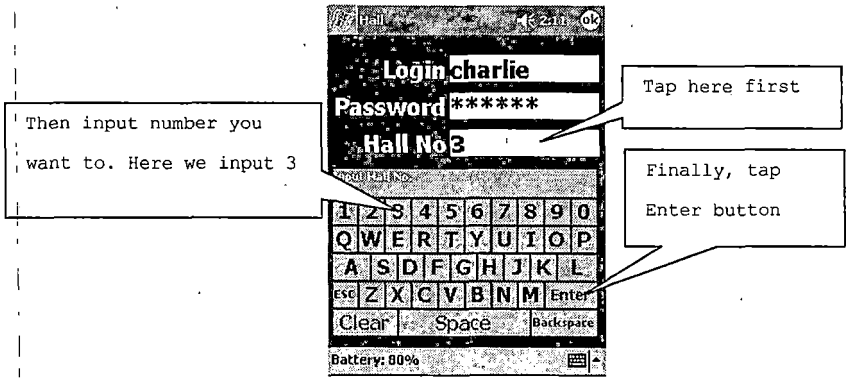


Figure 19. Input Hall Number

(2.9) If the inputting username, password (and hall number) isn't/aren't correct, the screen will show "Sorry, please relogin" and you have to repeat Step 2.1-2.8 to input correct username, password (and hall number)

Figure 20. Error in Login

(2.10) If the username, password are correct, your Pocket PC will load menu and other information

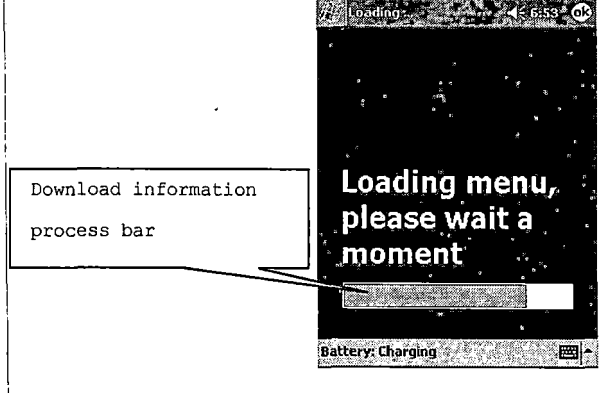


Figure 21. Download Procedure

(2.11) Table layout: after your Pocket PC loads menu, it will show the hall name which you serve and display the table layout of this hall

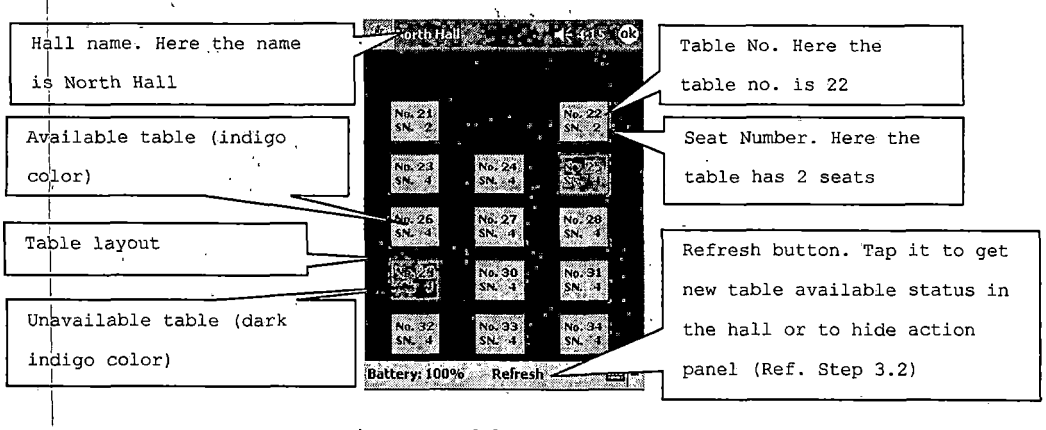


Figure 22. Table Layout

3. Tap the table which you want to hold for new coming customers, add items/order item, cancel hold, check out or show order item status

(3.1) Hold an available table for new coming customers

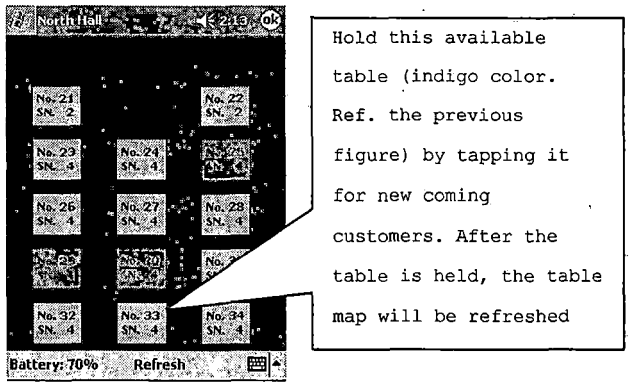


Figure 23. Hold a Table

(3.2)Add items/order items

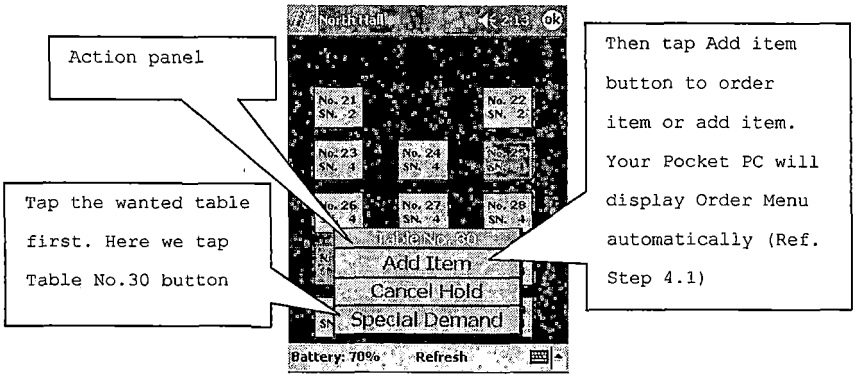


Figure 24. Add Item

(3.3)Cancel hold

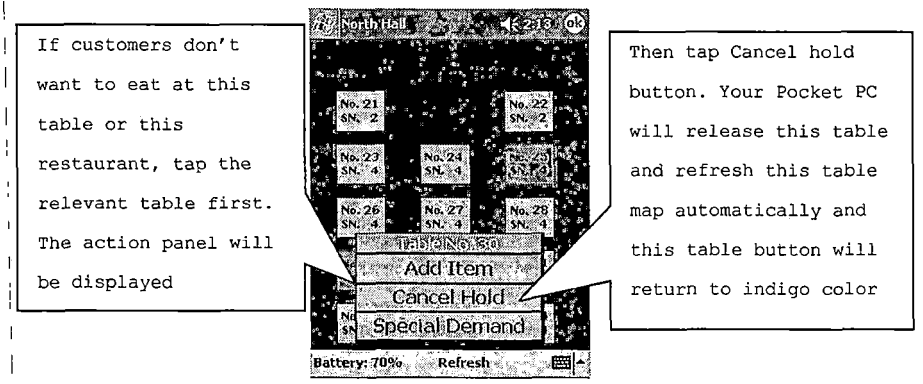


Figure 25. Cancel Hold

(3.4) Check out

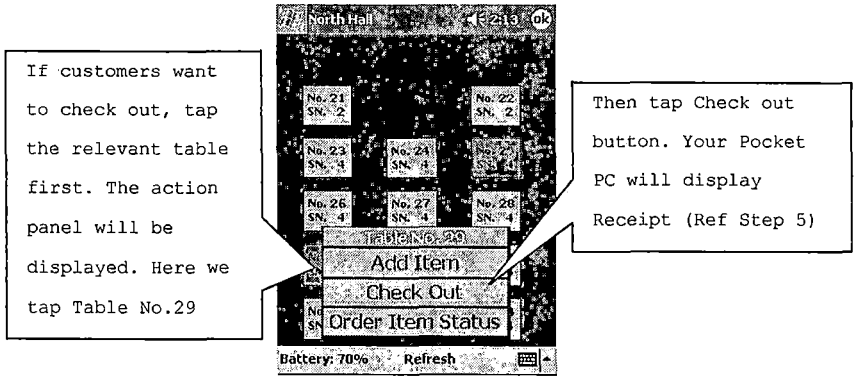


Figure 26. Check out

(3.5) Special Demand

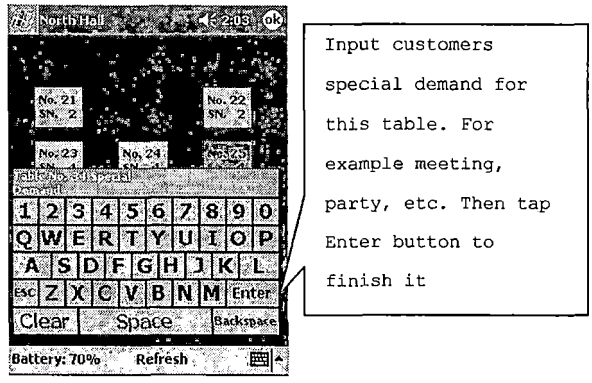


Figure 27. Input Special Demand

(3.6)Order Item Status

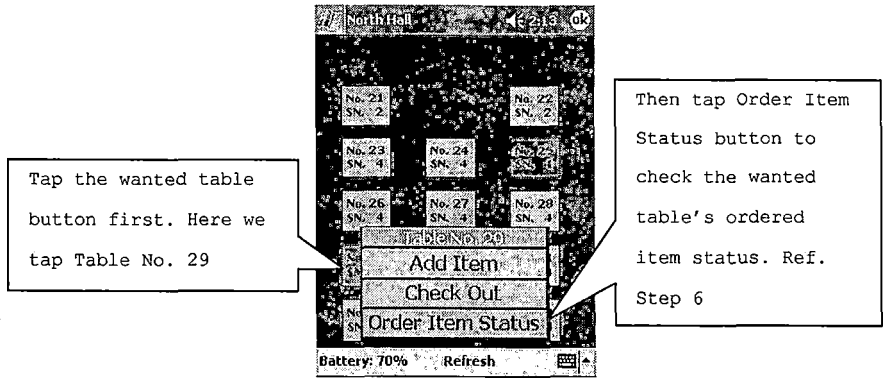


Figure 28. Show Ordered Items' Status

4. Add items: after you tap Add item button at Step 3.2 or Step 4.13, your Pocket PC will show menu automatically, and you can follow Step 4.2-4.6 to add items

(4.1)Order Menu

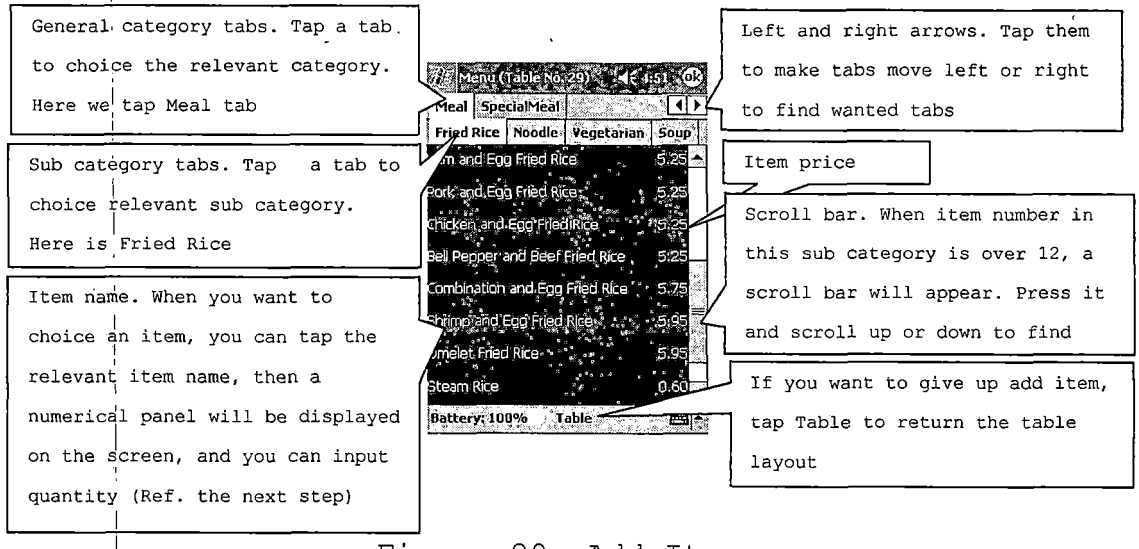


Figure 29. Add Items

(4.2) Choice item

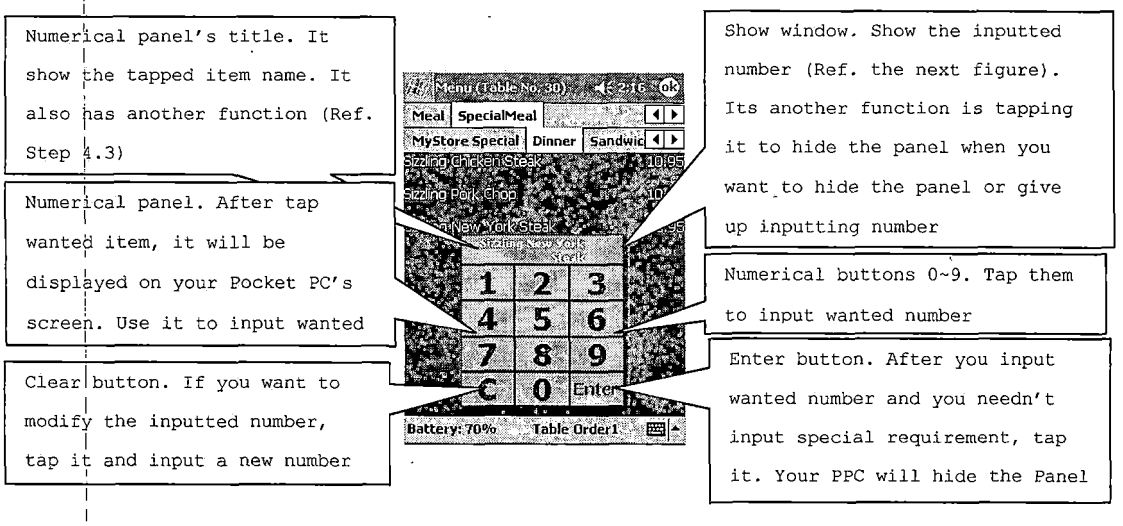


Figure 30. Numerical Panel

(4.3) Input quantity

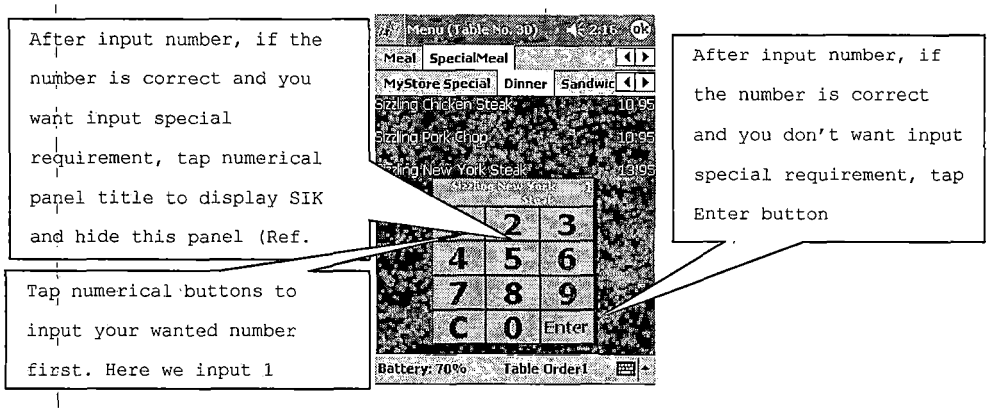


Figure 31. Input Quantity

(4.4)Modify input number

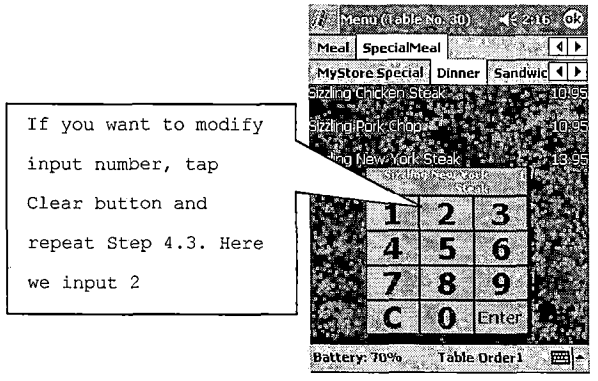


Figure 32. Modify Input Number

(4.5)Input special demand

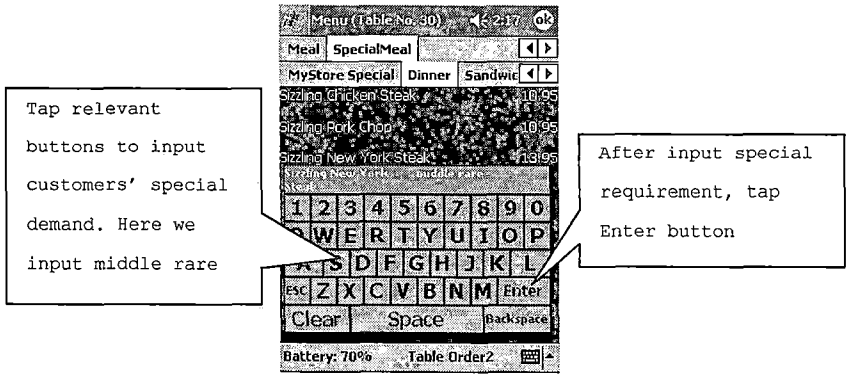


Figure 33. Input Special Demand

(4.6)Repeat Step 4.2-4.5 to finish adding item

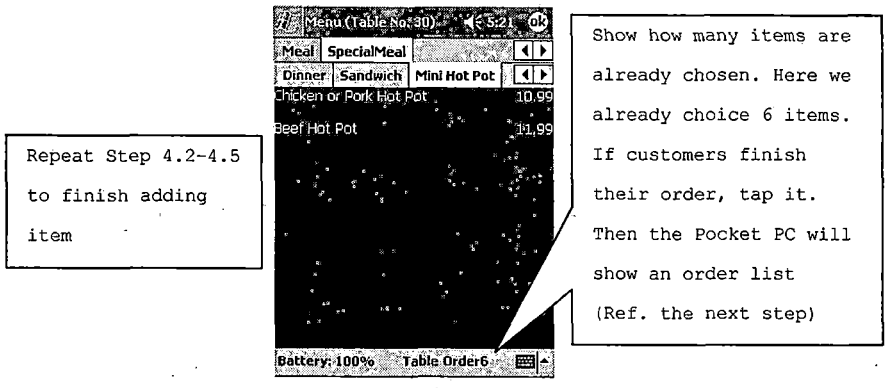


Figure 34. Add More Items

(4.7)Order List

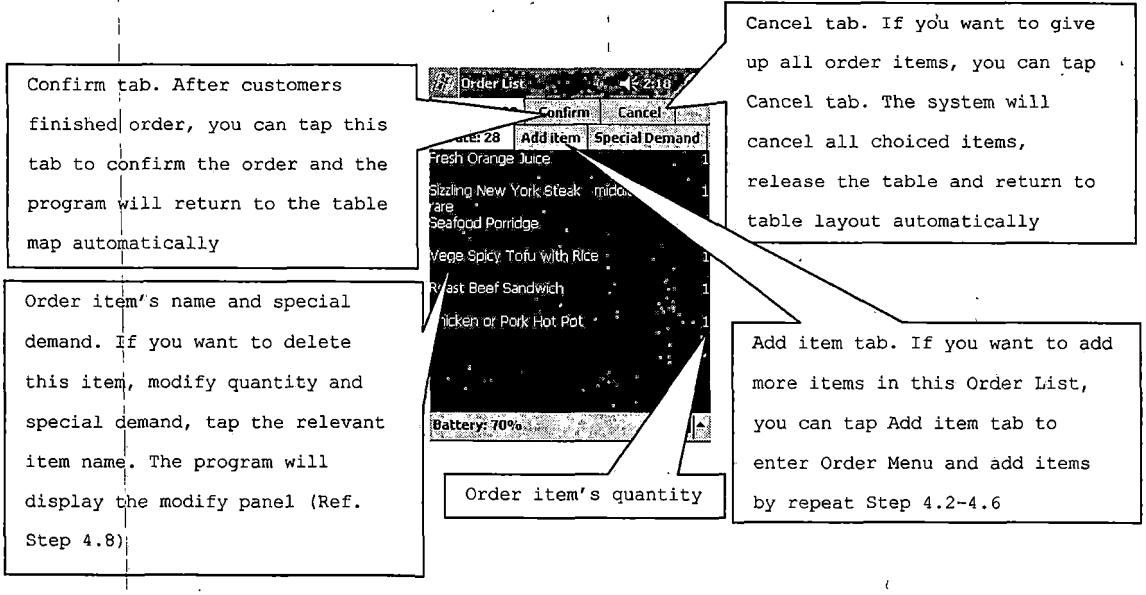


Figure 35. Show Order List

(4.8)Modify panel

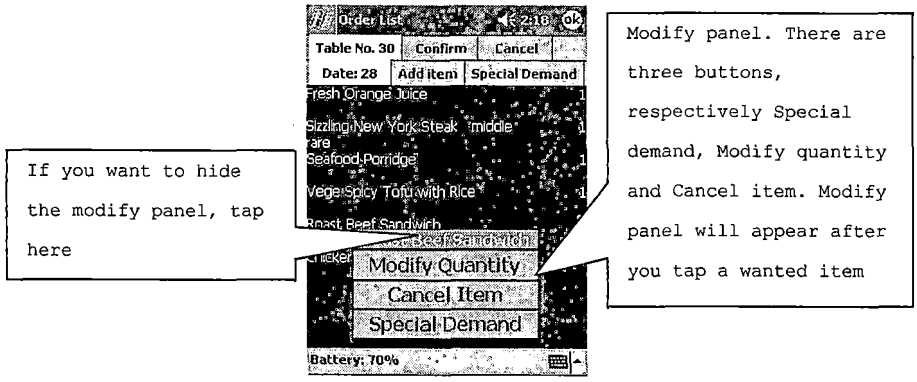


Figure 36. Modify Panel

(4.9)Delete chosen item

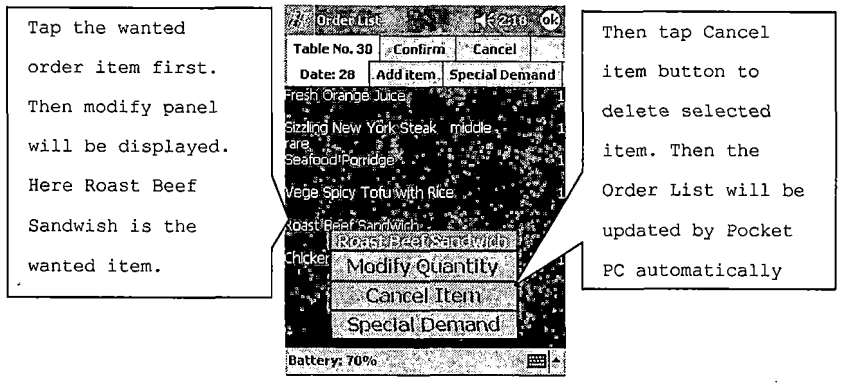
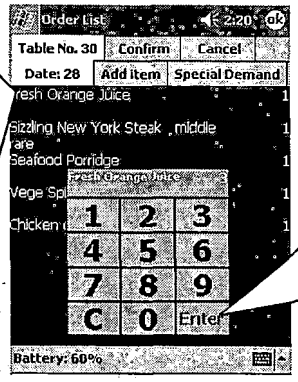


Figure 37. Delete Chosen Item

(4.10) Modify chosen items' quantity

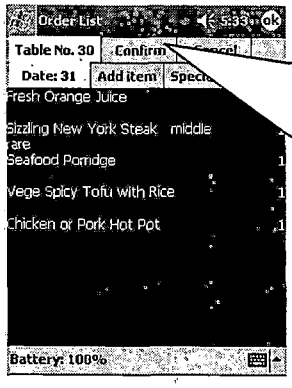
Tap the wanted order item's name first. After modify panel is displayed, tap Modify quantity button and the numerical panel will be displayed. Here Fresh Orange Juice is the wanted item.



Then input new number and tap Enter button. Here the number is 3 (Ref. the next figure)

Figure 38. Modify Chosen Item's Quantity

(4.11) Confirm the order



Tap Confirm tab to confirm the. Your Pocket PC will return the table map automatically. All order information will be sent to counter PC and will be diplayed in kitchen PC's waiting list.

Figure 39. Confirm the Order

(4.12) Cancel the order

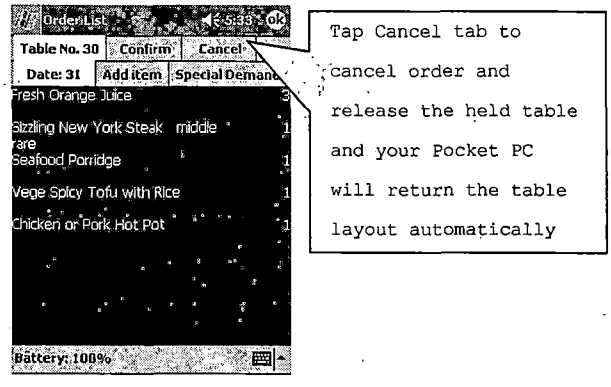


Figure 40. Cancel the Order

(4.13) Add more items in Order List

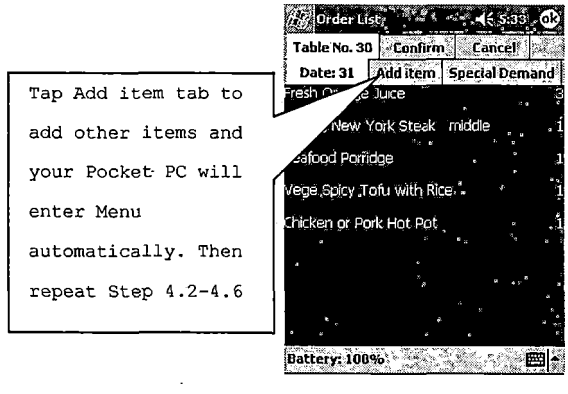


Figure 41. Add More Items in Order List

(4.14) Special demand for the table

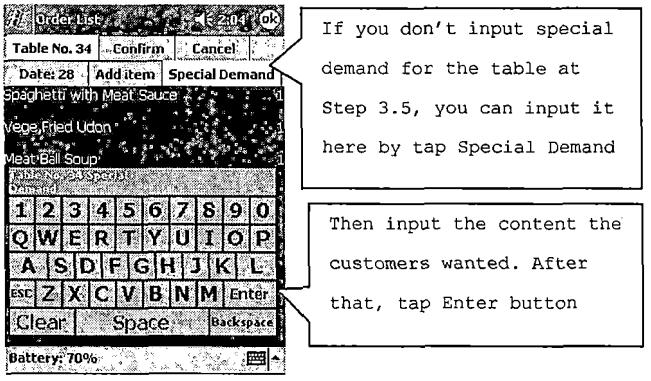


Figure 42. Special Demand for the Table

5. Check out

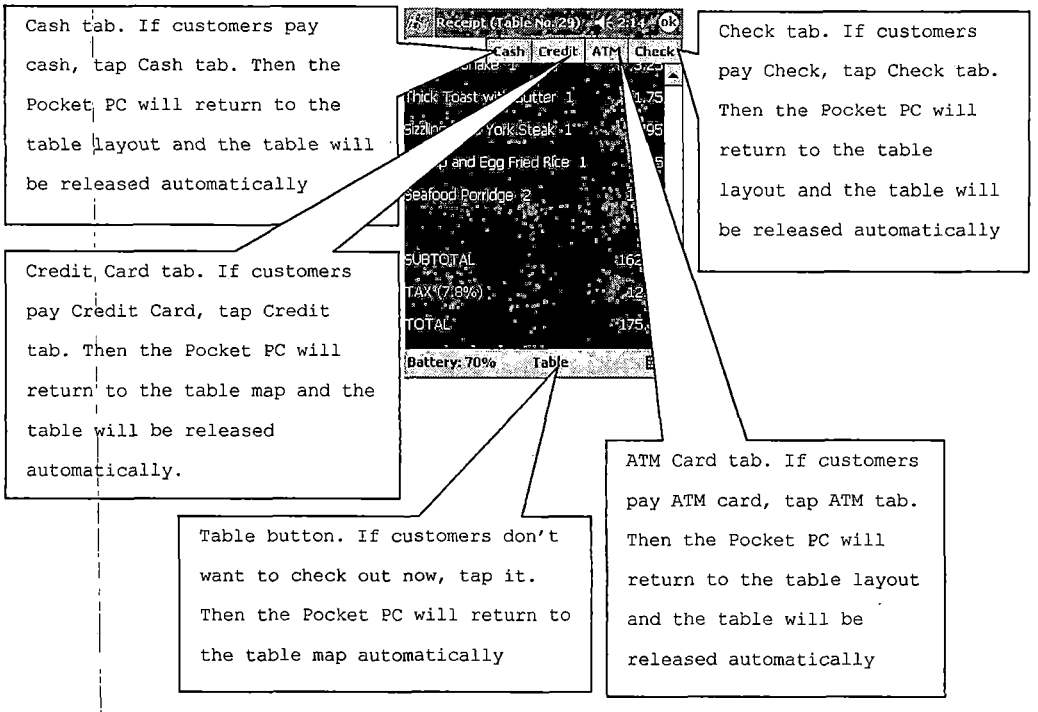


Figure 43. Check out

6. Ordered items' status

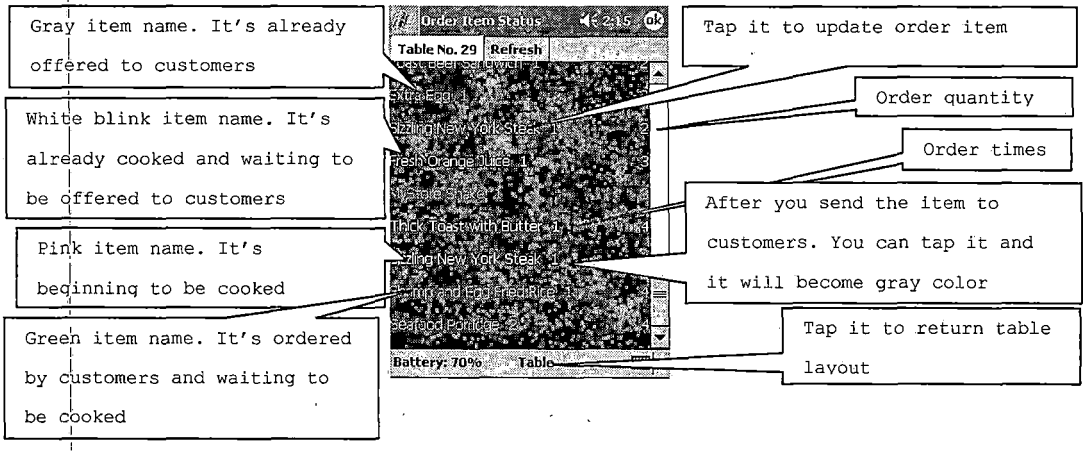


Figure 44. Show Ordered Items' Status

7. Exit mOrder Pocket PC side's program

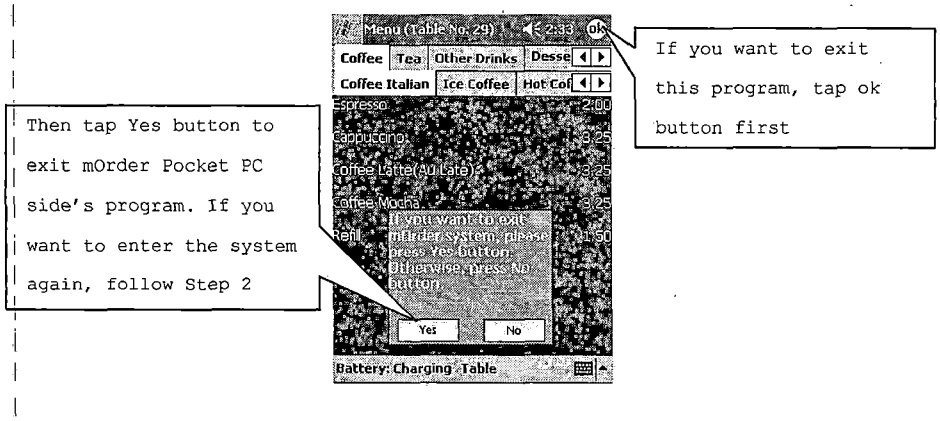


Figure 45. Exit mOrder Program

8. Alarm to take ready order items

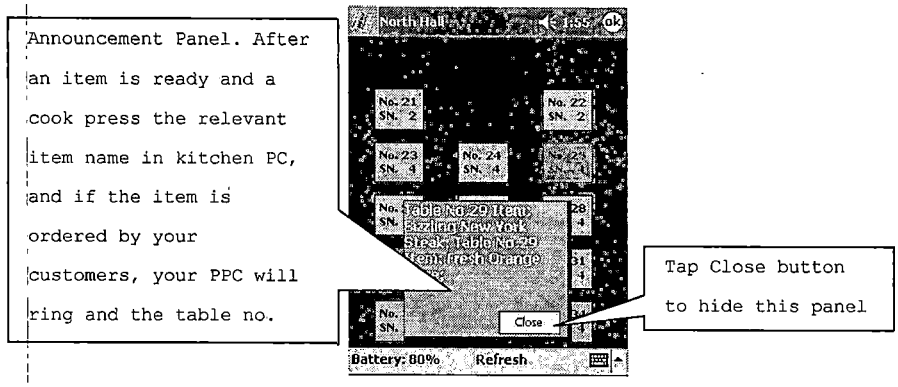


Figure 46. Show Ready Items' Information

Troubleshooting

1. If you walk far from the Access Point (AP), extend antenna or signal relay, your Pocket PC mayn't communicate with the counter PC. Walking close to AP, antenna or relay and tap Try again button

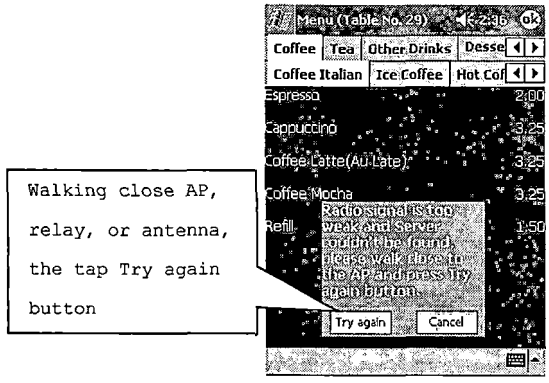


Figure 47. Network Problem

2. No Alarm Sound

(2.1) First Step

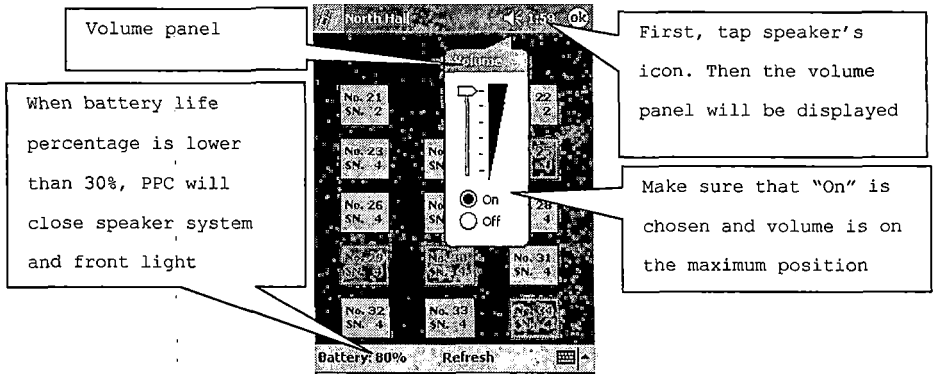


Figure 48. Adjust Sound Volume

(2.2) Second Step

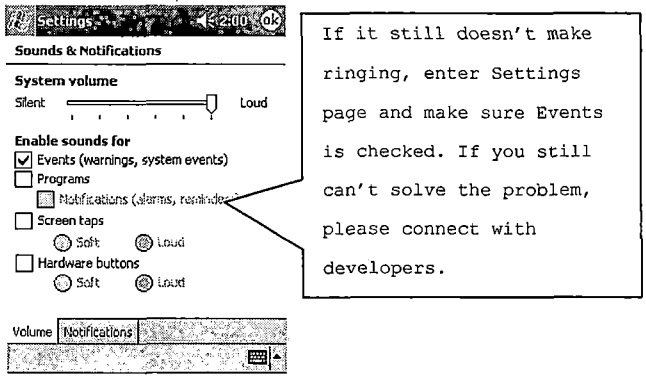


Figure 49. Setup Sound

3. Network Settings

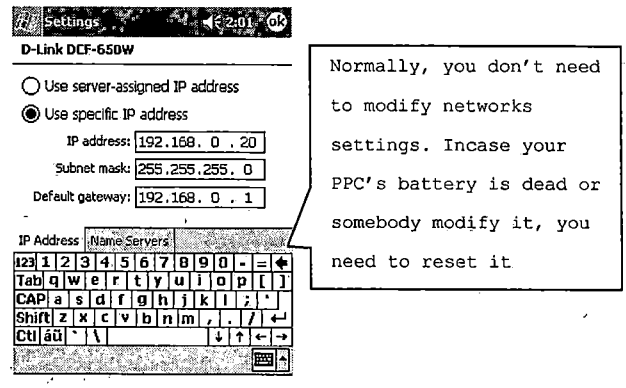


Figure 50. Network Setup

4. Time Settings

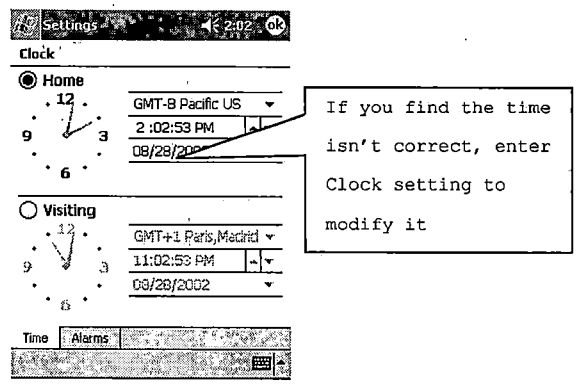


Figure 51. Time Setup

5. Recharge your Pocket PC's battery

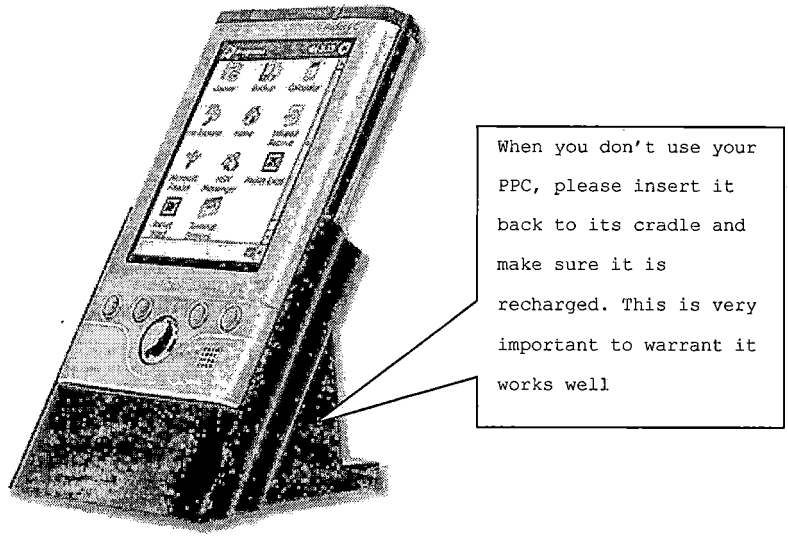


Figure 52. Recharge the Pocket PC

5.3 Kitchen Part Users Manual

Kitchen side's user manual is very simple. If you begin to cook a dish, just click or touch it in the waiting list. If you finish it, then click or touch it in the cooking list. When there are new orders, the program will prompt with voice, such as "There are 2 new orders!"

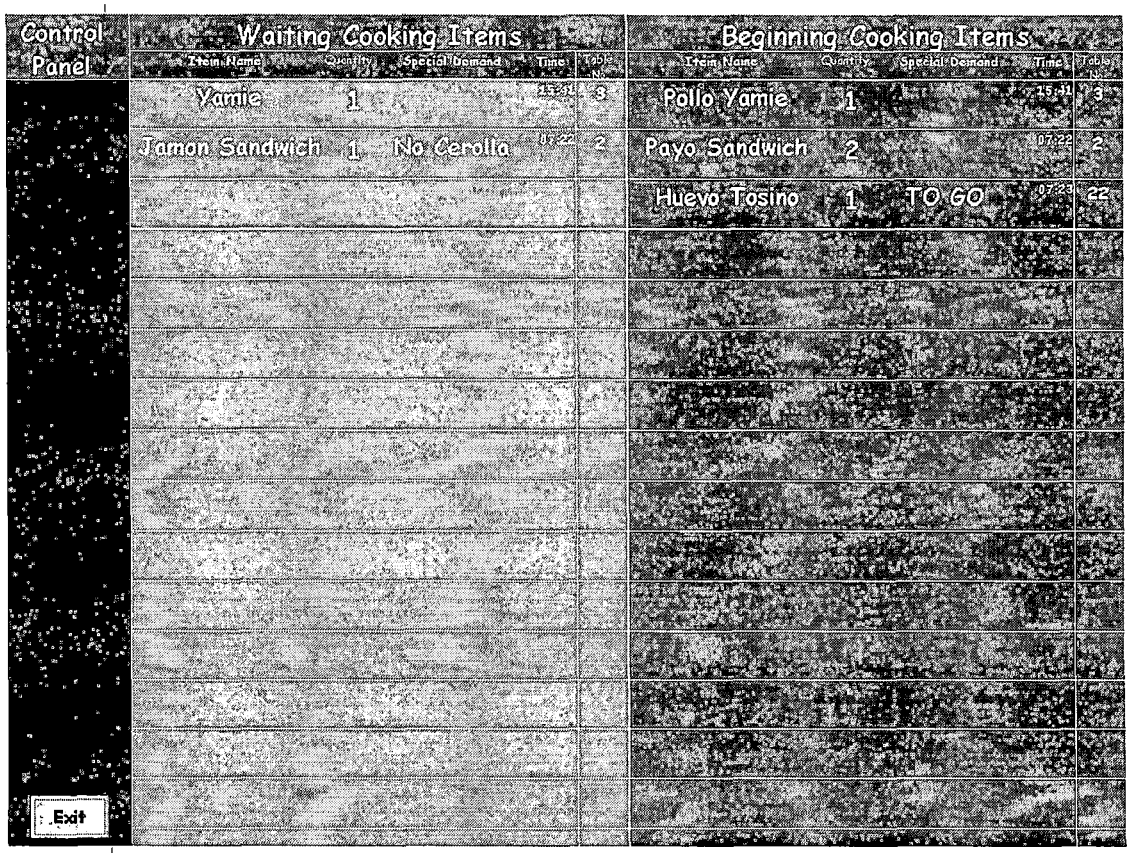


Figure 53. Kitchen Side Screenshot

CHAPTER SIX

CONCLUSIONS

1. mOrder-Client Food Service can improve service quality and efficiency of a restaurant because of using Wi-Fi and Pocket PC.

2. mOrder-Client Food Service has three parts: Pocket PC, Counter and Kitchen. Pocket PC part uses socket and GUI to send and get information from counter and kitchen and show the information in GUI style. Waiters/waitresses use these GUI controls to hold table, release table, order meal, check ordered meals' status and check out when customers finish their meal. Counter Side's task is retrieve/update/insert data from/in/into the database according to Pocket PC's requests via IIS. Kitchen part is an optional part. It can update ordered items' status and send the cooked item's information to Pocket PC.

3. Because the Pocket PC' part program retrieves battery and network information, and uses ring sound to alarm when it get the information from kitchen part, the project becomes more Commercial useful.

4. mOrder-Client Food Service needs some improvements at GUI, access pointer assigned IP address (DHCP), hardware setup (Refer Troubleshooting in Chapter FIVE), getting the table layout and menu's information, multilingual support

and languge switch function, and a commercial release package. If the improved version of mOrder-Client Food Service combines with the Ambol POS, it will become very good commercial software.

APPENDIX A

POCKET PC PART SOURCE CODE

```
‘’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’
‘Program Name:  mOrderPocketPC
‘Author:        Li Qiu
‘Date:          May 03, 2002
‘Version:       1.53
‘’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’

Option Explicit
Const MaxCategoryNumber As Integer = 10
Const MaxSubcategoryNumber As Integer = 10
Const MaxTableNumber As Integer = 50
Const MaxItemNumber As Integer = 54
Const IPPort As Integer = 80
Const IPPort2 As Long = 50000
Const IPPort2Counter As Long = 65001
Const ServerAddress As String = "192.168.0.10"
Const PPCIPAddress As String = "192.168.0.20"
Const ButtonMargin As Integer = 50
Const isfTop As Integer = 1460
Const ipStep As Integer = 268
Dim PocketPCIPAddress As String
Dim sHTML, sKitchen
Dim ASPFile As String
Dim str0, str1, str2, str3, str4, str5, str6, strp1, strp2, strp3, strp4, strp5 As String
Dim Location1, Location2 As Long
Dim i, j, k As Integer
Dim MaxOrderItemNumber As Integer

Dim StoreName As String
Dim HallName As String
Dim HallNumber As Integer
Dim PhoneNumber As String
Dim Address As String
Dim City As String
Dim State As String
Dim ZipCode As String
Dim TaxRate As String
Dim WaiterNumber As String
Dim WaiterWorkHallNumber As String
Dim CategoryName() As String
Dim TotalCategoryNumber As Integer
Dim SubcategoryName() As String
Dim TotalSubcategorynumber() As Integer
Dim ItemNumber() As Integer
Dim ItemName() As String
Dim ItemPrice() As String
Dim TotalSubitemNumber() As Integer
Dim Promotion1(), Promotion2(), Promotion3(), Promotion4(), Promotion5() As String
Dim TableNumber() As String
Dim TableAvailableStatus() As Boolean
Dim TableSeatNumber() As String
Dim TableXCoordinate() As String
Dim TableYCoordinate() As String
Dim TableWidth() As String
Dim TableLength() As String
Dim TotalTableNumber As Integer
Dim TableAvailableColor As Long
Dim TableUnavailableColor As Long
Dim TableCheckoutColor As Long
Dim TableWaiterNumber() As String
Dim TableReserve() As String
Dim TableNumberString As String
Dim MenuTableNumberString As String
Dim counterNumber As Integer
```

```
Dim OrderItemNumber() As Integer
Dim OrderItemName() As String
Dim OrderItemPrice() As String
Dim OrderItemQuantity() As Integer
Dim OrderItemSpecialRequisition() As String
Dim OrderItemSpecialDemand() As String
Dim OrderItemAddTimes() As Integer
Dim OrderItemAddMoney() As String
Dim OrderItemStatus() As String
Dim OrderNumber() As String
Dim OrderWaiterNumber() As String
Dim OrderSpecialRequisition As String
Dim OrderSpecialDemand As String
Dim OrderAddTimes As Integer
Dim AlreadyOrderItemNumber As Integer
Dim OrderTableIndex As Integer
Dim TotalOrderItemNumber As Integer
Dim SpecialDemand As String
Dim OrderMenuString As String
Dim ItemAddTimes As String
Dim OrderItemSeriesNumber() As String
Dim OrderItemPromotion() As String
Dim orderItemOffered As String

Dim Sum, Product, Tax, Total As Double
Dim StartSign, CheckoutSign As Integer

Dim lbItem As Label
Dim lbPrice As Label
Dim lbTable As CommandButton
Dim lbCaption As String
Dim lbTop As Integer
Dim lbLeft As Integer
Dim lbHeight As Integer
Dim lbWidth As Integer
Dim lbBackColor As Long
Dim lbForeColor As Long
Dim lbVisible As Boolean
Dim lbFontBold As Boolean
Dim lbFontSize As Integer
Dim lbFontName As String
Dim lbAlignment As Integer

Dim mbButton As MenuBarButton

Dim itemIndex As Integer

Dim Tabstrip1Status As Boolean
Dim Tabstrip2Status As Boolean
Dim HScroll1Status As Boolean
Dim VScroll1Status As Boolean
Dim MaxHeight As Integer
Dim maxWidth As Integer

Dim Tabstrip1Index As Integer
Dim Tabstrip2Index As Integer

Dim aTab

Dim ItemInitialTop As Integer
Dim ItemHeight As Integer
Dim ItemWidth As Integer
Dim ItemBackColor As Long
Dim ItemForeColor As Long
Dim ItemFontBold As Boolean
```

```
Dim ItemFontSize
Dim ItemFontName As String
Dim ItemTop As Integer
Dim itemsHeight As Integer
Dim ItemPriceWidth As Integer

Dim updateTableAvailableStatus As String
Dim keyboardTitleString As String

Dim readOrderItemDetailInfo As String

Dim itemOrderColor, itemCookingColor, itemReadyColor, itemOfferColor, itemBlinkColor As Long

Dim readyInfo, readyItemName, readyTableNumber As String
Dim sOrderNumber As String

Dim PowerStatus As Long ' pointer to SYSTEM_POWER_STATUS_EX structure
Dim LifePercent, oLifePercent, acOnlineStatus As Byte
Dim plPercent As String

Dim iProcess As Integer
Dim categoryNumber As Integer

Public Declare Function MessageBeep Lib "Coredll" (ByVal wType As Long) As Long
Public Declare Function PlaySound Lib "Coredll" Alias "PlaySoundW" (ByVal lpszName As String, ByVal hModule As Long,
ByVal dwFlags As Long) As Long

Public Declare Function AllocPointer Lib "VBPointers" Alias "VB_AllocPointer" (ByVal nSrcSizeInBytes As Long) As Long
Public Declare Function FreePointer Lib "VBPointers" Alias "VB_FreePointer" (ByVal Pointer As Long) As Long
Public Declare Function GetSystemPowerStatusEx Lib "Coredll" (ByVal PowerStatus As Long, ByVal Update As Long) As Long
Public Declare Function GetByteAt Lib "VBPointers" Alias "VB_GetByteAt" (ByVal Pointer As Long, ByVal OffsetInBytes As
Long) As Byte
'-----------------------------------------------------------------------------------------------------------

Private Sub AddButton_Click()
  ShowStatusFrame False
  If AddButton.Caption = "Add Item" Then
    HideTable
    TotalOrderItemNumber = 0
    If OrderNumber(OrderTableIndex) = "" Then
      AlreadyOrderItemNumber = 0
      MenuList MenuTableNumberString
    Else
      ASPFile = "ReadOrderItemNumber.asp"
      str1 = OrderNumber(OrderTableIndex)
      GetInfo
    End If
  ElseIf AddButton.Caption = "Modify Quantity" Then
    ShowNumPanelFrame True
  End If
End Sub

Private Sub AddTab(aTabStrip As TabStrip, aCaption As String)
  Set aTab = aTabStrip.Tabs.Add
  aTab.Caption = aCaption
  aTab.Key = aCaption
End Sub

Private Sub cSocket_Close()
  cSocket.Close
End Sub

Private Sub cSocket_Connect()
  cSocket.SendData "O" + sOrderNumber
End Sub
```

```
Private Sub cSocket_Error(ByVal number As Long, ByVal description As String)
  'MsgBox description
End Sub

Private Sub cSocket_SendComplete()
  cSocket.Close
End Sub
Private Sub GetInfo()
On Error Resume Next

  WinSock1.Connect
  If Err.number <> 0 Then
    'MsgBox "Err-conn: " & Err.Number & vbCrLf & Err.Description
    ShowMiscFrame "Connect"
  End If

  If ASPFile = "ReadItemInfo.asp" Or ASPFile = "ReadHallInfo.asp" Or ASPFile = "ReadOrderInfo.asp" Or ASPFile =
"ReadOrderItemDetailInfo.asp" Or ASPFile = "ReadOrderItemNumber.asp" Or ASPFile = "ReadAllTableInfo.asp" Or ASPFile =
"ReadUncheckoutOrder.asp" Then '1 Parameter
    str0 = "GET /mOrder/" + ASPFile + "?Parm1=" + str1 + vbCrLf + vbCrLf
  ElseIf ASPFile = "ReadWaiterInfo.asp" Or ASPFile = "UpdateAddTimes.asp" Or ASPFile = "UpdateOrderItemStatus.asp" Then
'2 Parameter
    str0 = "GET /mOrder/" + ASPFile + "?Parm1=" + str1 + "&Parm2=" + str2 + vbCrLf + vbCrLf
  ElseIf ASPFile = "Checkout.asp" Or ASPFile = "ReadWaiterHallInfo.asp" Then  '3 Parameter
    str0 = "GET /mOrder/" + ASPFile + "?Parm1=" + str1 + "&Parm2=" + str2 + "&Parm3=" + str3 + vbCrLf + vbCrLf
  ElseIf ASPFile = "UpdateTableAvailableStatus.asp" Or ASPFile = "Order.asp" Then
    str0 = "GET /mOrder/" + ASPFile + "?Parm1=" + str1 + "&Parm2=" + str2 + "&Parm3=" + str3 + "&Parm4=" + str4 +
vbCrLf + vbCrLf
  ElseIf ASPFile = "OrderItem.asp" Then
    str0 = "POST /mOrder/" & ASPFile & vbCrLf
    str0 = str0 & "Content-Type: "
    str0 = str0 & "application/x-www-form-urlencoded" & vbCrLf
    str0 = str0 & "Content-Length: " & Len(str1) & vbCrLf
    str0 = str0 & vbCrLf & str1
  Else ' No Parameter: ReadFoodCategory.asp, ReadItemInfo.asp, ReadStoreInfo.asp, ReadSubfoodCategory.asp
    str0 = "GET /mOrder/" + ASPFile + vbCrLf + vbCrLf
  End If
  WinSock1.SendData str0
  If Err.number <> 0 Then
    'MsgBox "Err-send: " & Err.Number & vbCrLf & Err.Description
    ShowMiscFrame "SendData"
  End If
End Sub
Private Sub ItemControl(anItem As Label, Index As Integer)
  itemIndex = Index
  If mOrderForm.Caption = MenuTableNumberString Then
    NumPanelTitleLabel.Caption = ItemName(Index, Tabstrip2Index, Tabstrip1Index)
    ShowNumPanelFrame True
  ElseIf mOrderForm.Caption = "Order List" Then
    NumPanelTitleLabel.Caption = anItem.Caption
    ShowStatusFrame True
  ElseIf mOrderForm.Caption = "Order Item Status" Then
    If anItem.ForeColor = itemReadyColor Or anItem.ForeColor = itemBlinkColor Then
      anItem.ForeColor = itemOfferColor
      ASPFile = "UpdateOrderItemStatus.asp"
      str1 = OrderItemSeriesNumber(itemIndex)
      str2 = orderItemOffered
      GetInfo
    End If
  End If
End Sub

Private Sub NumPanelControl(keyIndex As String)
  NumPanelInputLabel.Caption = NumPanelInputLabel.Caption + keyIndex
```

```
End Sub
Private Sub ReadFoodCategory(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11
  i = 0
  Do
    i = i + 1
    Location2 = InStr(Location1, sHTML, ",")
    CategoryName(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
  Loop While Location2 < StringLength
  TotalCategoryNumber = i

  ASPFile = "ReadSubfoodCategory.asp"
  GetInfo
End Sub

Private Sub ReadItem(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11
  i = 0
  j = 1

  Do
    Location2 = InStr(Location1, sHTML, ";")
    str1 = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ";")
    str2 = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ";")
    str3 = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ";")
    str4 = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    If i = 0 Then str6 = str4

    If str6 = str4 Then
      i = i + 1
    Else
      TotalSubitemNumber(j, categoryNumber) = i
      i = 1
      j = j + 1
    End If
    str6 = str4
    ItemNumber(i, j, categoryNumber) = str1
    ItemName(i, j, categoryNumber) = str2
    ItemPrice(i, j, categoryNumber) = str3
  Loop While Location2 < StringLength
  TotalSubitemNumber(j, categoryNumber) = i

  If categoryNumber = TotalCategoryNumber Then
    TableMap
  Else
    categoryNumber = categoryNumber + 1
    ASPFile = "ReadItemInfo.asp"
    str1 = CStr(categoryNumber)
    GetInfo
  End If
End Sub
Private Sub ReadOrderInfo(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11
```

```
  Location2 = InStr(Location1, sHTML, ",")
  str0 = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
  Location2 = InStr(Location1, sHTML, ",")
  TotalMoney = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
  Location2 = InStr(Location1, sHTML, ",")
  PayWay = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
  Location2 = InStr(Location1, sHTML, ",")
  BeginningTime = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
  Location2 = InStr(Location1, sHTML, ",")
  EndTime = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
  Location2 = InStr(Location1, sHTML, ",")
  str3 = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
  Location2 = InStr(Location1, sHTML, ",")
  SpecialDemand = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
  Location2 = InStr(Location1, sHTML, ",")
  ItemAddTimes = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
End Sub
Private Sub ReadOrderItemDetail(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11
  i = 0
  Do
    i = i + 1
    Location2 = InStr(Location1, sHTML, ";")
    OrderItemSeriesNumber(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
    Location2 = InStr(Location1, sHTML, ";")
    OrderItemNumber(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ";")
    OrderItemName(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
    Location2 = InStr(Location1, sHTML, ";")
    OrderItemPrice(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
    Location2 = InStr(Location1, sHTML, ";")
    OrderItemQuantity(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
    Location2 = InStr(Location1, sHTML, ";")
    OrderItemSpecialDemand(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
    Location2 = InStr(Location1, sHTML, ";")
    OrderItemAddMoney(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
    Location2 = InStr(Location1, sHTML, ";")
    OrderItemStatus(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ";")
    OrderItemAddTimes(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
    Location2 = InStr(Location1, sHTML, ";")
    OrderItemPromotion(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1
  Loop While Location2 < StringLength
  TotalOrderItemNumber = i
  If readOrderItemDetailInfo = "Check Out" Then
```

```
    ShowReceipt
  ElseIf readOrderItemDetailInfo = "Order Item Status" Then
    ShowOrderItemStatus
  End If
End Sub
Private Sub ReadOrderNumber(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11
  OrderNumber(OrderTableIndex) = Mid(sHTML, Location1, StringLength - Location1 + 1)
  ReadyWriteItemDetail
End Sub
Private Sub ReadOrderItemNumber(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11
  Location2 = InStr(Location1, sHTML, ",")
  AlreadyOrderItemNumber = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1
  ItemAddTimes = Mid(sHTML, Location1, StringLength - Location1 + 1)
  MenuList MenuTableNumberString
End Sub

Private Sub ReadStoreInfo(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11

  Location2 = InStr(Location1, sHTML, ",")
  StoreName = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1

  Location2 = InStr(Location1, sHTML, ",")
  PhoneNumber = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1

  Location2 = InStr(Location1, sHTML, ",")
  Address = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1

  Location2 = InStr(Location1, sHTML, ",")
  City = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1

  Location2 = InStr(Location1, sHTML, ",")
  State = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1

  Location2 = InStr(Location1, sHTML, ",")
  ZipCode = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1

  Location2 = InStr(Location1, sHTML, ",")
  TaxRate = Mid(sHTML, Location1, Location2 - Location1)
  Location1 = Location2 + 1

  ASPFile = "ReadFoodCategory.asp"
  GetInfo
End Sub
Private Sub ReadSubfoodCategory(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11
  str3 = "1"
  i = 0
  j = 1

  Do
    Location2 = InStr(Location1, sHTML, ",")
    str1 = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
```

```
    str2 = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    If str3 = str2 Then
      i = i + 1
    Else
      TotalSubcategorynumber(j) = i
      i = 1
      j = j + 1
    End If
    str3 = str2
    SubcategoryName(i, j) = str1
  Loop While Location2 < StringLength
  TotalCategoryNumber = j
  TotalSubcategorynumber(j) = i

  categoryNumber = 1
  ASPFile = "ReadItemInfo.asp"
  str1 = CStr(categoryNumber)
  GetInfo
End Sub

Private Sub ReadTableInfo(StringLength As Long)
  Location1 = InStr(230, sHTML, "private") + 11
  i = 0

  Do
    i = i + 1
    Location2 = InStr(Location1, sHTML, ",")
    TableNumber(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
    TableAvailableStatus(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
    TableSeatNumber(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
    TableXCoordinate(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
    TableYCoordinate(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
    TableWaiterNumber(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
    TableReserve(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
    TableWidth(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    Location2 = InStr(Location1, sHTML, ",")
    TableLength(i) = Mid(sHTML, Location1, Location2 - Location1)
    Location1 = Location2 + 1

    If TableSeatNumber(i) = "0" Then
```

```
        counterNumber = i
      End If
    Loop While Location2 < StringLength
    TotalTableNumber = i

    If StartSign = 0 Then
      StartSign = 1
      ASPFile = "ReadUncheckoutOrder.asp"
      str1 = WaiterNumber
      GetInfo
      LoginStatusLabel.Visible = False
      PSShape.Visible = False
      PFShape.Visible = False
    Else
      ShowTable
    End If
  End Sub
  Private Sub ReadUncheckoutOrder(StringLength As Long)
    Location1 = InStr(230, sHTML, "private") + 11
    If Location1 < StringLength Then
      i = 0
      Do
        i = i + 1
        Location2 = InStr(Location1, sHTML, ",")
        str1 = Mid(sHTML, Location1, Location2 - Location1)
        Location1 = Location2 + 1

        Location2 = InStr(Location1, sHTML, ",")
        str2 = Mid(sHTML, Location1, Location2 - Location1)
        Location1 = Location2 + 1

        For j = 1 To TotalTableNumber 'MaxTableNumber
          If TableNumber(j) = str2 Then
            OrderNumber(j) = str1
            OrderWaiterNumber(j) = WaiterNumber
          End If
        Next j
      Loop While Location2 < StringLength
    End If
    ShowTable
  End Sub

  Private Sub ReadWaiterInfo(StringLength As Long)
    If Mid(sHTML, StringLength - 6, StringLength) = "Correct" Then
      Location1 = InStr(230, sHTML, "private") + 11
      Location2 = InStr(Location1, sHTML, ",")
      WaiterWorkHallNumber = Mid(sHTML, Location1, Location2 - Location1)

      mOrderForm.Caption = "Loading ..."
      LoginStatusLabel.Caption = "Loading menu, please wait a moment"
      PFShape.Visible = True
      PSShape.Visible = True
      iProcess = 0
      PSShape.Width = 0
      ASPFile = "ReadHallInfo.asp"
      str1 = WaiterWorkHallNumber
      GetInfo
    Else
      LoginLabel.Caption = ""
      PasswordLabel.Caption = ""
      HallNoLabel.Caption = ""
      LoginFrame.Visible = True
      LoginStatusLabel.Caption = "Sorry, please relogin"
    End If
  End Sub
```

```
Private Sub ReadyWriteItemDetail()
  ASPFile = "OrderItem.asp"
  str1 = "OrderNumber=" + OrderNumber(OrderTableIndex) + "&AddTimes=" + ItemAddTimes

  For i = 1 To TotalOrderItemNumber
    If OrderItemQuantity(i) <> 0 Then
      str1 = str1 + "&ItemNumber=" + OrderItemNumber(i) + _
        "&Quantity=" + OrderItemQuantity(i) + _
        "&SpecialDemand=" + OrderItemSpecialDemand(i) + _
        "&AddMoney=" + OrderItemAddMoney(i) + _
        "&Promotion=" + OrderItemPromotion(i)
    End If
  Next i
  GetInfo
End Sub
Private Sub ShowCategory()
  TabStrip1.Tabs.Clear
  For i = 1 To TotalCategoryNumber
    AddTab TabStrip1, CategoryName(i)
  Next i
  TabStrip1.Visible = True

  TabStrip2.Tabs.Clear
  For i = 1 To TotalSubcategorynumber(1)
    AddTab TabStrip2, SubcategoryName(i, 1)
  Next i
  TabStrip2.Visible = True
End Sub
Private Sub ShowKeyboard(showStatus As Boolean)
  KeyboardInputLabel.Caption = ""
  SpecialDemand = ""
  KeyboardTitleLabel.Caption = keyboardTitleString
  'KeyboardFrame.Left = (Me.Width - KeyboardFrame.Width + VScroll1.Visible * VScroll1.Width) / 2
  KeyboardFrame.Top = VScroll1.Value + Me.Height - KeyboardFrame.Height - ButtonMargin
  KeyboardFrame.Visible = showStatus
End Sub
Private Sub ShowMiscFrame(aString As String)
  If aString = "Exit" Then
    MiscLabel.Caption = "If you want to exit mOrder system, please press Yes button. Otherwise, press No button."
    TryAgain.Caption = "Yes"
    TryAgain.Visible = True
    Cancel.Caption = "No"
  ElseIf aString = "Ready Item" Then
    MiscLabel.Caption = readyInfo
    Cancel.Caption = "Close"
    TryAgain.Visible = False
  Else
    MiscLabel.Caption = "Radio signal is too weak and Server couldn't be found, please walk close to the AP and press Try again
button."
    TryAgain.Caption = "Try again"
    TryAgain.Visible = True
    Cancel.Caption = "Cancel"
  End If

  MiscFrame.Left = (Me.Width - MiscFrame.Width + VScroll1.Width * VScroll1.Visible) / 2
  MiscFrame.Top = VScroll1.Value + Me.Height - MiscFrame.Height - ButtonMargin
  MiscFrame.Visible = True
End Sub

Private Sub ShowNumPanelFrame(showStatus As Boolean)
  NumPanelInputLabel.Caption = ""
  NumPanelFrame.Left = (Me.Width - NumPanelFrame.Width + VScroll1.Visible * VScroll1.Width) / 2
  NumPanelFrame.Top = VScroll1.Value + Me.Height - NumPanelFrame.Height - ButtonMargin
  NumPanelFrame.Visible = showStatus
End Sub
```

```
Private Sub ShowOrder()
On Error Resume Next
  HideItem
  HidePrice
  HideTable

  j = 0
  For i = 1 To TotalOrderItemNumber
    If OrderItemQuantity(i) = 0 Then j = j + 1
  Next i

  MaxHeight = ItemHeight * (TotalOrderItemNumber - j) + ItemInitialTop
  VScroll1.Height = Me.Height - TabStrip1.Height - TabStrip2.Height
  VScroll1.Top = TabStrip2.Top + TabStrip2.Height
  If MaxHeight <= Me.Height Then
    VScroll1.Visible = False
    VScroll1.Value = 0  'There is a bug
  Else
    VScroll1.Max = MaxHeight - Me.Height
    VScroll1.LargeChange = VScroll1.Max / (MaxHeight / Me.Height)
    VScroll1.SmallChange = Item01.Height
    VScroll1.Visible = True
  End If

  j = 0
  For i = 1 To TotalOrderItemNumber
    If OrderItemQuantity(i) = 0 Then
      j = j + 1
    Else
      SelectItem i
      lbCaption = OrderItemName(i) & "   " & OrderItemSpecialRequisition(i)
      lbTop = ItemInitialTop + (i - 1 - j) * ItemHeight
      lbAlignment = vbLeftJustify
      SetItemProperties

      SelectPrice i
      lbCaption = OrderItemQuantity(i)
      lbAlignment = vbRightJustify
      SetPriceProperties
    End If
  Next i

  SetBatteryMenu
End Sub

Private Sub ShowOrderItemStatus()
  HideFrames
  HideItem
  HidePrice

  mOrderForm.Caption = "Order Item Status"
  TabStrip1.Tabs.Clear
  AddTab TabStrip1, TableNumberString
  AddTab TabStrip1, "Refresh"
  AddTab TabStrip1, "Check out"
  TabStrip1.Visible = True
  TabStrip1.Enabled = True
  TabStrip2.Visible = False

  MaxHeight = ItemHeight * TotalOrderItemNumber + TabStrip1.Height
  VScroll1.Value = 0
  VScroll1.Height = Me.Height - TabStrip1.Height
  VScroll1.Top = TabStrip1.Height
  If MaxHeight <= Me.Height Then
```

```
        VScroll1.Visible = False
    Else
        VScroll1.Max = MaxHeight - Me.Height
        VScroll1.LargeChange = VScroll1.Max / (MaxHeight / Me.Height)
        VScroll1.SmallChange = Item01.Height
        VScroll1.Visible = True
    End If

    j = 0
    For i = 1 To TotalOrderItemNumber
        If OrderItemQuantity(i) = 0 Then
            j = j + 1
        Else
            Select Case OrderItemStatus(i)
                Case "0"
                    ItemForeColor = itemOrderColor
                Case "1"
                    ItemForeColor = itemCookingColor
                Case "2"
                    ItemForeColor = itemReadyColor
                Case "3"
                    ItemForeColor = itemOfferColor
            End Select
            SelectItem i
            lbCaption = OrderItemName(i) & "  " & OrderItemQuantity(i)
            lbTop = TabStrip1.Height + (i - 1 - j) * ItemHeight
            lbAlignment = vbLeftJustify
            SetItemProperties

            SelectPrice i
            lbCaption = OrderItemAddTimes(i)
            lbAlignment = vbRightJustify
            SetPriceProperties
        End If
    Next i
    ItemForeColor = itemReadyColor
    SetTableMenu
End Sub
Private Sub ShowReceipt()
    HideFrames
    HideItem
    HidePrice

    mOrderForm.Caption = "Receipt (" + TableNumberString + ")"
    TabStrip1.Tabs.Clear
    AddTab TabStrip1, " Payment "
    AddTab TabStrip1, "Cash"
    AddTab TabStrip1, "Credit"
    AddTab TabStrip1, "ATM"
    AddTab TabStrip1, "Check"
    TabStrip1.Visible = True
    TabStrip1.Enabled = True
    TabStrip2.Visible = False

    MaxHeight = ItemHeight * (TotalOrderItemNumber + 4) + TabStrip1.Height
    VScroll1.Value = 0
    VScroll1.Height = Me.Height - TabStrip1.Height
    VScroll1.Top = TabStrip1.Height
    If MaxHeight <= Me.Height Then
        VScroll1.Visible = False
    Else
        VScroll1.Max = MaxHeight - Me.Height
        VScroll1.LargeChange = VScroll1.Max / (MaxHeight / Me.Height)
        VScroll1.SmallChange = Item01.Height
        VScroll1.Visible = True
```

```
    End If

    Sum = 0#
    j = 0
    For i = 1 To TotalOrderItemNumber
      SelectItem i
      If OrderItemQuantity(i) = 0 Then
        j = j + 1
      Else
        Product = OrderItemQuantity(i) * OrderItemPrice(i)
        Sum = Sum + Product
        lbCaption = OrderItemName(i) & "  " & OrderItemQuantity(i)
        lbTop = TabStrip1.Height + (i - 1 - j) * ItemHeight
        lbAlignment = vbLeftJustify
        SetItemProperties

        SelectPrice i
        lbCaption = FormatNumber(Product, 2)
        lbAlignment = vbRightJustify
        SetPriceProperties
      End If
    Next i
    Sum = FormatNumber(Sum, 2)

'-----------------------Show Sum-------------------------
    SelectItem TotalOrderItemNumber + 1
    lbCaption = "SUBTOTAL"
    lbTop = TabStrip1.Height + (TotalOrderItemNumber - j + 1) * ItemHeight
    lbAlignment = vbLeftJustify
    SetItemProperties

    SelectPrice TotalOrderItemNumber + 1
    lbCaption = Sum
    lbAlignment = vbRightJustify
    SetPriceProperties
'-----------------------Show Tax-------------------------
    SelectItem TotalOrderItemNumber + 2
    Tax = FormatNumber(Sum * TaxRate, 2)
    lbTop = TabStrip1.Height + (TotalOrderItemNumber - j + 2) * ItemHeight
    lbCaption = "TAX" + " (" + CStr(TaxRate * 100) + "%)"
    lbAlignment = vbLeftJustify
    SetItemProperties

    SelectPrice TotalOrderItemNumber + 2
    lbCaption = Tax
    lbAlignment = vbRightJustify
    SetPriceProperties
'-----------------------Show Total-------------------------
    SelectItem TotalOrderItemNumber + 3
    Total = FormatNumber(Sum * (1# + TaxRate), 2)
    lbCaption = "TOTAL"
    lbTop = TabStrip1.Height + (TotalOrderItemNumber - j + 3) * ItemHeight
    lbAlignment = vbLeftJustify
    SetItemProperties

    SelectPrice TotalOrderItemNumber + 3
    lbCaption = Total
    lbAlignment = vbRightJustify
    SetPriceProperties
'-----------------------------------------------------
    SetTableMenu
  End Sub

  Private Sub ShowStatusFrame(showStatus As Boolean)
    If showStatus Then
```

```
    If mOrderForm.Caption = HallName Then
      StatusTitleLabel.Caption = TableNumberString "Table No. " + TableNumber(OrderTableIndex)
      AddButton.Caption = "Add Item"
      PromotionButton.Caption = "Coupon"
      If OrderNumber(OrderTableIndex) = "" Then
        CancelButton.Caption = "Cancel Hold"
        DemandButton.Caption = "Special Demand"
        ItemAddTimes = ""
      Else
        CancelButton.Caption = "Check Out"
        DemandButton.Caption = "Order Item Status"
        ItemAddTimes = "1"
      End If
    ElseIf mOrderForm.Caption = "Order List" Then
      StatusTitleLabel.Caption = OrderItemName(itemIndex)
      AddButton.Caption = "Modify Quantity"
      CancelButton.Caption = "Cancel Item"
      DemandButton.Caption = "Special Demand"
      PromotionButton.Caption = "Promtion"
    End If
    StatusFrame.Left = (Me.Width - StatusFrame.Width + VScroll1.Visible * VScroll1.Width) / 2
    StatusFrame.Top = VScroll1.Value + Me.Height - StatusFrame.Height - ButtonMargin
  End If
  StatusFrame.Visible = showStatus
End Sub

Private Sub ShowTable()
  If mOrderForm.Caption <> "Loading ..." Or mOrderForm.Caption <> HallName Then
    HideItem
    HidePrice
    TabStrip1.Visible = False
    TabStrip2.Visible = False
    VScroll1.Visible = False
  End If
  mOrderForm.Caption = HallName
  HideFrames

  For i = 1 To TotalTableNumber
    SelectTable i
    If Len(TableNumber(i)) = 1 Then
      str0 = "No. 0"
    Else
      str0 = "No. "
    End If
    If i = counterNumber Then
      lbCaption = ""
    Else
      lbCaption = str0 & TableNumber(i) & " SN.  " & TableSeatNumber(i)
    End If
    If TableAvailableStatus(i) Then
      lbBackColor = TableAvailableColor
    Else
      lbBackColor = TableUnavailableColor
    End If
    lbVisible = True
    lbTop = TableYCoordinate(i)
    lbLeft = TableXCoordinate(i)
    lbWidth = 660   'TableLength(i, hallindex)
    If i = counterNumber Then
      lbHeight = TableLength(i)  'TableWidth(i, hallindex)
    Else
      lbHeight = 450  'TableWidth(i, hallindex)
    End If
    ShowTableButton
  Next i
```

```
  If updateTableAvailableStatus = "holdTable" Or updateTableAvailableStatus = "cancelHold" Then
    EnabledTables
  End If
  updateTableAvailableStatus = ""
  SetRefreshMenu
End Sub

Private Sub StatusTitleLabel_Click()
  ShowStatusFrame False
End Sub

Private Sub TableControl(aTable As CommandButton, tableIndex As Integer)
  If StatusFrame.Visible Or MiscFrame.Visible Or KeyboardFrame.Visible Or tableIndex = counterNumber Then
  Else
    OrderTableIndex = tableIndex
    If Len(TableNumber(OrderTableIndex)) = 1 Then
      TableNumberString = "Table No. 0" + TableNumber(OrderTableIndex)
    Else
      TableNumberString = "Table No. " + TableNumber(OrderTableIndex)
    End If
    MenuTableNumberString = "Menu (" + TableNumberString + ")"

    If aTable.BackColor = TableAvailableColor Then 'hold table
      MenuBar1.Controls.Clear
      DisabledTables
      updateTableAvailableStatus = "holdTable"
      ASPFile = "UpdateTableAvailableStatus.asp"
      str1 = WaiterWorkHallNumber
      str2 = "False"
      str3 = TableNumber(OrderTableIndex)
      str4 = WaiterNumber
      GetInfo
      TableWaiterNumber(OrderTableIndex) = WaiterNumber
      OrderNumber(OrderTableIndex) = ""
      'MenuList menutablenumberstring
    Else
      If OrderWaiterNumber(OrderTableIndex) = WaiterNumber Then
        ShowStatusFrame True
      ElseIf TableWaiterNumber(OrderTableIndex) = WaiterNumber Then
        ShowStatusFrame True
      End If
    End If
  End If
End Sub
Private Sub TryAgain_Click()
  If TryAgain.Caption = "Try again" Then
    MiscFrame.Visible = False
    GetInfo
  Else
    App.End
  End If
End Sub

Private Sub WinSock2_Close()
  WinSock2.Close
  WinSock2.Listen
End Sub

Private Sub WinSock2_ConnectionRequest()
  'If WinSock2.State <> sckClosed Then
  '   WinSock2.Close
  'End If
  WinSock2.Accept
End Sub
```

```
Private Sub WinSock2_DataArrival(ByVal bytesTotal As Long)
   WinSock2.GetData sKitchen

   Location1 = 1
   Location2 = InStr(Location1, sKitchen, ",")
   readyItemName = Mid(sKitchen, Location1, Location2 - Location1)
   Location1 = Location2 + 1

   Location2 = InStr(Location1, sKitchen, ",")
   readyTableNumber = Mid(sKitchen, Location1, Location2 - Location1)
   Location1 = Location2 + 1
   readyInfo = readyInfo + "Table No." + readyTableNumber + " Item: " + readyItemName + "; "
   ShowMiscFrame "Ready Item"
   For i = 1 To 50
      MessageBeep (&H30&)
      'MessageBeep (&H10&)
   Next i
   'If WinSock2.State <> sckClosed Then
   '   WinSock2.Close
   '   WinSock2.Listen
   'End If
End Sub

Private Sub WinSock2_Error(ByVal number As Long, ByVal description As String)
   WinSock2.Close
   WinSock2.Listen
End Sub
```

APPENDIX B

COUNTER PART SOURCE CODE

```
<% @Language = "VBScript" %>
<%
'-----------------------------------------------------------------------------
' File:        Checkout.asp
' Purpose:     Demonstrate retrieving a recordset
' Version:     1.0
' Author:      Li Qiu
' Date:        May 22, 2002
'-----------------------------------------------------------------------------
Dim sSQLQuery, oConn, rs
Set oConn = Server.CreateObject("ADODB.Connection")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
oConn.Execute "UPDATE Orders SET Total =" & Request.QueryString("Parm2") &_
            ", PayWay='" & Request.QueryString("Parm3") & "', EndDate='" & Date &_
                  "',EndTime='" & time & "' WHERE OrderNumber =" & Request.QueryString("Parm1")
%>

<% @Language = "VBScript" %>
<%
'-----------------------------------------------------------------------------
' File:        Order.asp
' Purpose:     Calling a stored procedure from an ASP to insert a record.
' Version:     1.0
'-----------------------------------------------------------------------------
Dim sSQLQuery, cmd, ordertime, oConn, rs
On Error Resume Next
Set oConn = Server.CreateObject("ADODB.Connection")
Set cmd = Server.CreateObject("ADODB.Command")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
Set cmd.ActiveConnection = oConn
ordertime = now
cmd.CommandText = "INSERT INTO Orders (TableNumber, BeginningDate, BeginningTime," &_
          "WaiterNumber,SpecialDemand,AddTimes,PocketPCIPAddress) VALUES (" &_
          Request.QueryString("Parm1") & ",'" & date & "','"& time() & "','" &_
          Request.QueryString("Parm2") &"','" & Request.QueryString("Parm3") & "',1,'" &_
          Request.QueryString("Parm4") & "')"
cmd.CommandType = adCmdText
cmd.Execute
sSQLQuery="SELECT MAX(OrderNumber) AS NewOrder FROM Orders WHERE TableNumber=" &_
          Request.QueryString("Parm1") & " AND WaiterNumber='" &_
          Request.QueryString("Parm2") & "'"
Set rs = Server.CreateObject("ADODB.Recordset")
rs.Open sSQLQuery, oConn
Response.Write(rs("NewOrder"))
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------------------------
' File:       OrderItem.asp
' Purpose:    Calling a stored procedure from an ASP to insert a record.
' Version:    1.0
' ---------------------------------------------------------------------------
Dim sSQLQuery, cmd, totalNumber, orderNumber, addTimes
On Error Resume Next
Set oConn = Server.CreateObject("ADODB.Connection")
Set cmd = Server.CreateObject("ADODB.Command")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
Set cmd.ActiveConnection = oConn
cmd.CommandType = adCmdText
orderNumber=Request.Form("OrderNumber")
addTimes=Request.Form("AddTimes")
For totalNumber = 1 to Request.Form("ItemNumber").Count
        cmd.CommandText = "INSERT INTO OrderItem(OrderNumber, ItemNumber, Quantity, SpecialDemand,
                AddMoney,Status, AddTimes, Promotion, AddDate, AddTime) VALUES (" & orderNumber & "," &
                Request.Form("ItemNumber")(totalNumber) & "," & Request.Form("Quantity")(totalNumber) & ", '" &
                Request.Form("SpecialDemand")(totalNumber) & "', " &_
                Request.Form("AddMoney")(totalNumber) & ", 0," & addTimes & "," &_
                Request.Form("Promotion")(totalnumber) & ",'" & date & "','"&time & "')"
        cmd.Execute
Next
cmd.CommandText = "UPDATE Orders SET AddTimes="+addTimes+" WHERE OrderNumber="_
        +orderNumber
cmd.Execute
%>


<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:       ReadAllTableInfo.asp
' Purpose:    Retrieving a Recordset of Table Information
' Version:    1.0
' Author:     Li Qiu
' Date:       May 22, 2002
' ---------------------------------------------------------
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT TableNumber, AvailableStatus, SeatNumber, XCoordinate, " &_
        YCoordinate, WaiterNumber,ReserveTable " & ",Width, Length " &_
        "FROM TABLES WHERE HallNumber =" & Request.QueryString("Parm1")
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\morder\mOrderDatabase.mdb"
rs.Open sSQLQuery, oConn
Do Until rs.EOF
        Response.Write(rs("TableNumber") & ","_
                & rs("AvailableStatus") & "," _
                & rs("SeatNumber") & ", " _
                & rs("XCoordinate") & "," _
                & rs("YCoordinate") & "," _
                & rs("WaiterNumber") & "," _
                & rs("ReserveTable") & "," _
                & rs("Width") & "," _
                & rs("Length") & "," _
                )
        rs.MoveNext
Loop
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:        ReadFoodCategory.asp
' Purpose:     Demonstrate retrieving a recordset
' Version:     1.0
' Author:      Li Qiu
' Date:        May 22, 2002
' ---------------------------------------------------------
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT CategoryName FROM FOODCATEGORY"
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
Source=I:\Project\DataBase\Test.mdb"
rs.Open sSQLQuery, oConn
Do Until rs.EOF
        Response.Write(rs("CategoryName") & ",")
        rs.MoveNext
Loop
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:        ReadHallInfo.asp
' Purpose:     Calling a stored procedure from an ASP to insert a record.
' Version:     1.0
' ---------------------------------------------------------
Dim sSQLQuery, cmd, ordertime, oConn, rs
On Error Resume Next
Set oConn = Server.CreateObject("ADODB.Connection")
Set cmd = Server.CreateObject("ADODB.Command")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\mOrder\mOrderDatabase.mdb"
Set cmd.ActiveConnection = oConn
ordertime = now
sSQLQuery="SELECT HallName FROM Hall " & "WHERE HallNumber= " &_
        Request.QueryString("Parm1")
Set rs = Server.CreateObject("ADODB.Recordset")
rs.Open sSQLQuery, oConn
Response.Write(rs("HallName"))
%>
```

```
<% @Language = "VBScript" %>
<%
'-----------------------------------------------------
' File:        ReadItemInfo.asp
' Purpose:     Demonstrate retrieving a recordset
' Version:     1.0
' Author:      Li Qiu
' Date:        May 22, 2002
'-----------------------------------------------------
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT ItemNumber, ItemName, Price, ITEM.SubcategoryNumber " &_
            "FROM ITEM, SUBFOODCATEGORY " &_
            "WHERE ITEM.SubcategoryNumber=SUBFOODCATEGORY.SubcategoryNumber " & _
            AND CategoryNumber=" & Request.QueryString("Parm1") & " "&_
            "ORDER BY ITEM.SubcategoryNumber, ItemNumber"
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
rs.Open sSQLQuery, oConn
Do Until rs.EOF
            Response.Write(rs("ItemNumber") & ";"_
                        & rs("ItemName") & ";" _
                        & rs("Price") & ";"_
                        & rs("SubcategoryNumber") & ";"_
                        )
            rs.MoveNext
Loop
%>

<% @Language = "VBScript" %>
<%
'-----------------------------------------------------
' File:        ReadOrderInfo.asp
' Purpose:     Calling a stored procedure from an ASP to insert a record.
' Version:     1.0
'-----------------------------------------------------
Dim sSQLQuery, cmd, ordertime, oConn, rs
On Error Resume Next
Set oConn = Server.CreateObject("ADODB.Connection")
Set cmd = Server.CreateObject("ADODB.Command")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
Set cmd.ActiveConnection = oConn
ordertime = now
sSQLQuery="SELECT TableNumber, Total, PayWay, BeginningTime, EndTime, WaiterNumber," &_
            SpecialDemand,AddTimes " & "FROM ORDERS WHERE OrderNumber=" & _
            Request.QueryString("Parm1")
Set rs = Server.CreateObject("ADODB.Recordset")
rs.Open sSQLQuery, oConn
Response.Write(rs("TableNumber") & "," &_
            rs("Total") & "," &_
            rs("PayWay") & "," &_
            rs("BeginningTime") & "," &_
            rs("EndTime") & "," &_
            rs("WaiterNumber") & "," &_
            rs("SpecialDemand") & "," &_
            rs("AddTimes") & ",")
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:       ReadOrderItemDetailInfo.asp
' Purpose:    Demonstrate retrieving a recordset
' Version:    1.0
' Author:     Li Qiu
' Date:       May 22, 2002
' ---------------------------------------------------------
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT SeriesNumber, ITEM.ItemNumber as ItemNo, ItemName, Price, " & _
            Quantity, SpecialDemand, " &_
            " AddMoney, Status, AddTimes, Promotion FROM ORDERITEM, ITEM " &_
            "WHERE OrderNumber=" & Request.QueryString("Parm1") &_
            "AND ORDERITEM.ItemNumber = ITEM.ItemNumber"
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
rs.Open sSQLQuery, oConn
Do Until rs.EOF
            Response.Write(rs("SeriesNumber") & ";"_
                        & rs("ItemNo") & ";"_
                        & rs("ItemName") & ";"_
                        & rs("Price") & ";"_
                        & rs("Quantity") & ";"_
                        & rs("SpecialDemand") & ";"_
                        & rs("AddMoney") & ";"_
                        & rs("Status") & ";"_
                        & rs("AddTimes") & ";"_
                        & rs("Promotion") & ";")
            rs.MoveNext
Loop
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:       ReadOrderItemNumber.asp
' Purpose:    Demonstrate retrieving a recordset
' Version:    1.0
' Author:     Li Qiu
' Date:       May 22, 2002
' ---------------------------------------------------------
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT COUNT(OrderNumber) as TotalNumber, MAX(AddTimes) as MAT "&_
            "FROM ORDERITEM WHERE OrderNumber=" & Request.QueryString("Parm1")
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
rs.Open sSQLQuery, oConn
Response.Write(rs("TotalNumber") & "," & rs("MAT"))
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:        ReadStoreInfo.asp
' Purpose:     Calling a stored procedure from an ASP to insert a record.
' Version:     1.0
' ---------------------------------------------------------
Dim sSQLQuery, cmd, ordertime, oConn, rs
On Error Resume Next
Set oConn = Server.CreateObject("ADODB.Connection")
Set cmd = Server.CreateObject("ADODB.Command")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
Set cmd.ActiveConnection = oConn
ordertime = now
sSQLQuery="SELECT StoreName, PhoneNumber, Address, City, State, ZipCode, TaxRate "&_
          "FROM STORE"
Set rs = Server.CreateObject("ADODB.Recordset")
rs.Open sSQLQuery, oConn
Response.Write(rs("StoreName") & "," &_
          rs("PhoneNumber") & "," &_
          rs("Address") & "," &_
          rs("City") & "," &_
          rs("State") & "," &_
          rs("ZipCode") & "," &_
          rs("TaxRate") & ",")
%>


<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:        ReadSubfoodCategory.asp
' Purpose:     Demonstrate retrieving a recordset
' Version:     1.0
' Author:      Li Qiu
' Date:        May 22, 2002
' ---------------------------------------------------------
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT SubcategoryName, CategoryNumber FROM SUBFOODCATEGORY"
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
Source=I:\Project\DataBase\Test.mdb"
          rs.Open sSQLQuery, oConn
          Do Until rs.EOF
                    Response.Write(rs("SubcategoryName") & ","_
                              & rs("CategoryNumber") & ",")
                    rs.MoveNext
          Loop
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:       ReadUncheckoutOrder.asp
' Purpose:    Calling a stored procedure from an ASP to insert a record.
' Version:    1.0
' ---------------------------------------------------------
Dim sSQLQuery, cmd, ordertime, oConn, rs
On Error Resume Next
Set oConn = Server.CreateObject("ADODB.Connection")
Set cmd = Server.CreateObject("ADODB.Command")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
Set cmd.ActiveConnection = oConn
ordertime = now
sSQLQuery="SELECT OrderNumber, TableNumber FROM ORDERS " &_
          "WHERE Total IS NULL AND WaiterNumber ='" & Request.QueryString("Parm1")&"'"
Set rs = Server.CreateObject("ADODB.Recordset")
rs.Open sSQLQuery, oConn
Do Until rs.EOF
          Response.Write(rs("OrderNumber") & "," & rs("TableNumber") & "," )
          rs.MoveNext
Loop
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:       ReadWaiterHallInfo.asp
' Purpose:    Demonstrate retrieving a recordset
' Version:    1.0
' Author:     Li Qiu
' Date:       May 22, 2002
' ---------------------------------------------------------
On Error Resume Next
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT HALL.HallNumber AS hn FROM WAITER,HALL WHERE WaiterNumber ='" &_
          Request.QueryString("Parm1") +"' AND Password ='" + _
          Request.QueryString("Parm2")+ "' AND HALL.HallNumber="+_
          Request.QueryString("Parm3")
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
rs.Open sSQLQuery, oConn
Response.Write(rs("hn") & ",Correct")
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:       ReadWaiterInfo.asp
' Purpose:    Demonstrate retrieving a recordset
' Version:    1.0
' Author:     Li Qiu
' Date:       May 22, 2002
' ---------------------------------------------------------
On Error Resume Next
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT HallNumber FROM WAITER WHERE WaiterNumber ='" &_
            Request.QueryString("Parm1") +_
            "' AND Password ='" + Request.QueryString("Parm2")+"'"
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
rs.Open sSQLQuery, oConn
Response.Write(rs("HallNumber") & ",Correct")
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:       UpdateAddTimes.asp
' Purpose:    Demonstrate retrieving a recordset
' Version:    1.0
' Author:     Li Qiu
' Date:       May 22, 2002
' ---------------------------------------------------------
Dim sSQLQuery, oConn, rs
Set oConn = Server.CreateObject("ADODB.Connection")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
oConn.Execute "UPDATE Orders SET AddTimes =" & Request.QueryString("Parm2") &_
              " WHERE OrderNumber =" & Request.QueryString("Parm1")
%>
```

```
<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:        UpdateOrderItemStatus.asp
' Purpose:     Demonstrate retrieving a recordset
' Version:     1.0
' Author:      Li Qiu
' Date:        May 22, 2002
' ---------------------------------------------------------
Dim sSQLQuery, oConn, rs
Set oConn = Server.CreateObject("ADODB.Connection")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
oConn.Execute "UPDATE OrderItem SET Status =" & Request.QueryString("Parm2") &_
              " WHERE SeriesNumber =" & Request.QueryString("Parm1")
%>


<% @Language = "VBScript" %>
<%
' ---------------------------------------------------------
' File:        UpdateTableAvailableStatus.asp
' Purpose:     Retrieving a recordset and update Table Information
' Version:     1.0
' Author:      Li Qiu
' Date:        May 22, 2002
' ---------------------------------------------------------
Dim sSQLQuery, oConn, rs
sSQLQuery = "SELECT TableNumber, AvailableStatus, SeatNumber, XCoordinate, " &_
            "YCoordinate, WaiterNumber,ReserveTable " &_
            "FROM TABLES WHERE HallNumber =" & Request.QueryString("Parm1")
Set oConn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")
oConn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\morder\mOrderDatabase.mdb"
oConn.Execute "UPDATE Tables SET AvailableStatus =" & Request.QueryString("Parm2") &_
            ", WaiterNumber ='" & Request.QueryString("Parm4") &_
            "' WHERE TableNumber =" & Request.QueryString("Parm3")
%>
```

APPENDIX C

KITCHEN PART SOURCE CODE

```
‘’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’
‘ Program:     Kitchen Side Program
‘ Date:        May. 09, 2002
‘Author:       Li Qiu
‘Version:      1.12
‘’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’’

Const cInterval As Integer = 16 '12 for 1280*1024
Const maxItemNumber As Integer = 16 '12
Const PPCIPPort As Long = 50000
Const defaultPPCIPAddress As String = "192.168.0.20"
Const CounterPCIPAddress As String = "192.168.0.10"
Const CounterPCIPPort As Long = 60000
Const DATAPATH = "Peedy.acs"
Const playAgent As Boolean = True
Const tlChange As Integer = 130
Const margin As Integer = 25

Dim i, j, k As Integer

Dim wbFrameWidth As Long
Dim cFrameWidth As Integer

Dim nlWidth, qlWidth, splWidth, tlWidth, tnlWidth As Integer
Dim nlLeft, qlLeft, splLeft, tlLeft, tnlLeft As Integer

Dim wlTop, wlWidth, wlHeight As Integer
Dim lInterval As Integer

Dim sSQLQuery, dbEngine As String
Dim oConn, rs

Dim wSeriesNumber() As String
Dim wItemName() As String
Dim wQuantity() As String
Dim wSpecialDemand() As String
Dim wAddMoney() As String
Dim wTableNumber() As String
Dim wWaiterNumber() As String
Dim wPPCIPAddress() As String
Dim wAddTime() As String
Dim wChItemName() As String

Dim bSeriesNumber() As String
Dim bItemName() As String
Dim bQuantity() As String
Dim bSpecialDemand() As String
Dim bAddMoney() As String
Dim bTableNumber() As String
Dim bWaiterNumber() As String
Dim bPPCIPAddress() As String
Dim bChItemName() As String

Dim wItemNumber As Integer
Dim wItemNumberOld As Integer

Dim strMessage As String

Dim ESC, Cutter

Dim Peedy As IAgentCtlCharacterEx
Dim strlength As Integer

Dim scMessage As String

Dim speakString As String
```

```
Dim counterNumber As Integer

Private Sub BeginningItemShow()
  For i = 1 To maxItemNumber
    wlTop = bfTitle.Height + (i - 1) * (wlHeight + cInterval)

    bnLabel(i).Top = wlTop
    bnLabel(i).Left = nlLeft
    bnLabel(i).Width = nlWidth - margin / 2
    bnLabel(i).Height = wlHeight
    bnLabel(i).Visible = True

    bqLabel(i).Top = wlTop
    bqLabel(i).Left = qlLeft
    bqLabel(i).Width = qlWidth - margin
    bqLabel(i).Height = wlHeight
    bqLabel(i).Visible = True

    bspLabel(i).Top = wlTop
    bspLabel(i).Left = splLeft
    bspLabel(i).Width = splWidth - margin
    bspLabel(i).Height = wlHeight
    bspLabel(i).Visible = True

    btLabel(i).Top = wlTop
    btLabel(i).Left = tlLeft
    btLabel(i).Width = tlWidth
    btLabel(i).Height = wlHeight
    btLabel(i).Visible = True

    btnLabel(i).Top = wlTop
    btnLabel(i).Left = tnlLeft
    btnLabel(i).Width = tnlWidth
    btnLabel(i).Height = wlHeight
    btnLabel(i).Visible = True
  Next i
End Sub

Private Sub bItemUpdate(bIndex As Integer)
On Error Resume Next
  If bnLabel(bIndex).Caption <> "" Then
'     speakString = "\emp\Hi," + bWaiterNumber(bIndex) + _
'         ",Table \pau=100\Number \pit=200\" + bTableNumber(bIndex) + _
'         "'s \pit=80\order item,\pau=200\\spd=60" + bItemName(bIndex) + _
'         ",\spd=120\ \pau=100\is \pit=50\ready! \spd=180\" + _
'         "Please come to the kitchen to take it!"
    speakString = "\emp\Hi, Table \pau=100\Number \pit=200\" + bTableNumber(bIndex) + _
        "'s \pit=80\order item,\pau=200\\spd=60" + bChItemName(bIndex) + _
        ",\spd=120\ \pau=100\is \pit=50\ready!"
    'If playAgent Then
    '   Peedy.Show
    '   Peedy.Speak
    '   Peedy.Hide
    'End If
    sSQLQuery = "UPDATE OrderItem SET Status=2 WHERE SeriesNumber=" + bSeriesNumber(bIndex)
    oConn.Open dbEngine
    rs.Open sSQLQuery, oConn
    oConn.Close

    Winsock1.RemoteHost = bPPCIPAddress(bIndex)
    Winsock1.RemotePort = PPCIPPort
    Call Winsock1.Connect
    strMessage = bItemName(bIndex) + "," + bTableNumber(bIndex) + ","

    Call CounterSocket.Connect
```

```
        scMessage = speakString

        ReadBeginningItem
    End If
End Sub

Private Sub CounterSocket_Close()
    Call CounterSocket.Close
End Sub

Private Sub CounterSocket_Connect()
    Call CounterSocket.SendData(scMessage)
End Sub

Private Sub CounterSocket_SendComplete()
    Call CounterSocket.Close
End Sub

Private Sub ExitButton_Click()
    ExitFrame.Visible = True
    No.SetFocus

    'If playAgent Then Peedy.Show
    'Peedy.Speak "\spd=150\If you don't want to exit this program, press No Button. Don't \Pau=100\exit \Pau=100\it
\pau=100\when \pau=100\the \Pau=100\restaurant is running!"
    'Peedy.Hide
End Sub

Public Function FixedLengthString(aString As String, anInteger As String)
    stringLength = Len(aString)
    If stringLength >= anInteger Then
        FixedLengthString = Mid(aString, 1, anInteger)
    Else
        FixedLengthString = aString
        stringLength = anInteger - stringLength
        For i = 1 To stringLength
            If anInteger > 8 Then
                FixedLengthString = FixedLengthString + " "
            Else
                FixedLengthString = " " + FixedLengthString
            End If
        Next i
    End If
End Function

Private Sub Form_Load()
    ReDim wSeriesNumber(maxItemNumber)
    ReDim wItemName(maxItemNumber)
    ReDim wQuantity(maxItemNumber)
    ReDim wSpecialDemand(maxItemNumber)
    ReDim wAddMoney(maxItemNumber)
    ReDim wTableNumber(maxItemNumber)
    ReDim wWaiterNumber(maxItemNumber)
    ReDim wPPCIPAddress(maxItemNumber)
    ReDim wAddTime(maxItemNumber)
    ReDim wChItemName(maxItemNumber)

    ReDim bSeriesNumber(maxItemNumber)
    ReDim bItemName(maxItemNumber)
    ReDim bQuantity(maxItemNumber)
    ReDim bSpecialDemand(maxItemNumber)
    ReDim bAddMoney(maxItemNumber)
    ReDim bTableNumber(maxItemNumber)
    ReDim bWaiterNumber(maxItemNumber)
    ReDim bPPCIPAddress(maxItemNumber)
```

```
ReDim bChItemName(maxItemNumber)

Move 0, 0
cFrameWidth = ControlFrame.Width

For i = 1 To maxItemNumber
    Load wnLabel(i)
    Load wqLabel(i)
    Load wspLabel(i)
    Load wtLabel(i)
    Load wtnLabel(i)

    Load bnLabel(i)
    Load bqLabel(i)
    Load bspLabel(i)
    Load btLabel(i)
    Load btnLabel(i)
Next i

wlHeight = 1040
lInterval = 0

Set oConn = CreateObject("ADODB.Connection")
Set rs = CreateObject("ADODB.Recordset")
dbEngine = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=\\Counter\morder\mOrderDatabase.mdb"

ESC = Chr$(27)
Cutter = Chr$(100)

wItemNumber = 1
Agent1.Characters.Load "Peedy", DATAPATH
Set Peedy = Agent1.Characters("Peedy")
Peedy.LanguageID = &H409

If playAgent Then Peedy.Show
Peedy.Speak "\Spd=150\Welcome to use \spd=100\mOrder System \spd=150\Kitchen \Pit=50\side!"
Peedy.Hide

CounterSocket.RemoteHost = CounterPCIPAddress
CounterSocket.RemotePort = CounterPCIPPort
ReadCounterNumber
ReadWaitingItem
ReadBeginningItem

End Sub
Private Sub Form_Terminate()
  Call Winsock1.Close
End Sub
Private Sub ReadBeginningItem()
On Error Resume Next
  For i = 1 To maxItemNumber
    bnLabel(i).Caption = ""
    bqLabel(i).Caption = ""
    bspLabel(i).Caption = ""
    btLabel(i).Caption = ""
    btnLabel(i).Caption = ""

    bSeriesNumber(i) = ""
    bItemName(i) = ""
    bQuantity(i) = ""
    bSpecialDemand(i) = ""
    bAddMoney(i) = ""
    bTableNumber(i) = ""
    bWaiterNumber(i) = ""
```

```
    bPPCIPAddress(i) = ""
    bChItemName(i) = ""
  Next i

  sSQLQuery = "SELECT SeriesNumber,ItemName,
SpItemName,Quantity,OrderItem.CookingSpecialDemand,AddMoney," + _
          "AddTime, TableNumber, WaiterNumber, PocketPCIPAddress " + _
          "FROM OrderItem,Item,Orders " + _
          "WHERE Status=1 AND OrderItem.ItemNumber=Item.ItemNumber AND " + _
          "Orders.OrderNumber=OrderItem.OrderNumber AND CookingSign=True"
  oConn.Open dbEngine
  rs.Open sSQLQuery, oConn

  i = 0
  Do Until rs.EOF
    i = i + 1
    If i <= maxItemNumber Then
      bSeriesNumber(i) = rs("SeriesNumber")
      bItemName(i) = rs("SpItemName")
      bTableNumber(i) = rs("TableNumber")
      bWaiterNumber(i) = rs("WaiterNumber")
      bPPCIPAddress(i) = rs("PocketPCIPAddress")
      bChItemName(i) = rs("ItemName")

      bnLabel(i).Caption = rs("SpItemName")
      bqLabel(i).Caption = rs("Quantity")
      If rs("TableNumber") = counterNumber Then
        bspLabel(i).Caption = "TO GO " + rs("CookingSpecialDemand")
      Else
        bspLabel(i).Caption = rs("CookingSpecialDemand")
      End If
      btLabel(i).Caption = Mid(rs("AddTime"), 1, 5)
      btnLabel(i).Caption = FormatNumber(rs("AddMoney"), 2, vbTrue)
    End If
    rs.movenext
  Loop
  oConn.Close
End Sub

Private Sub ReadCounterNumber()
  sSQLQuery = "SELECT TableNumber FROM Tables WHERE SeatNumber='0'"
  oConn.Open dbEngine
  rs.Open sSQLQuery, oConn
  counterNumber = rs("TableNumber")
  oConn.Close
End Sub

Private Sub ReadWaitingItem()
On Error Resume Next
  For i = 1 To maxItemNumber
    wnLabel(i).Caption = ""
    wqLabel(i).Caption = ""
    wspLabel(i).Caption = ""
    wtLabel(i).Caption = ""
    wtnLabel(i).Caption = ""

    wSeriesNumber(i) = ""
    wItemName(i) = ""
    wQuantity(i) = ""
    wSpecialDemand(i) = ""
    wAddMoney(i) = ""
    wTableNumber(i) = ""
    wWaiterNumber(i) = ""
    wPPCIPAddress(i) = ""
    wAddTime(i) = ""
```

```
        wChItemName(i) = ""
    Next i

    sSQLQuery = "SELECT SeriesNumber,ItemName,
SpItemName,Quantity,OrderItem.CookingSpecialDemand,AddMoney," + _
            "AddTime, TableNumber, WaiterNumber, PocketPCIPAddress " + _
            "FROM OrderItem,Item,Orders " + _
            "WHERE Status=0 AND OrderItem.ItemNumber=Item.ItemNumber AND " + _
            "Orders.OrderNumber=OrderItem.OrderNumber AND CookingSign=True"
    oConn.Open dbEngine
    rs.Open sSQLQuery, oConn

    i = 0
    Do Until rs.EOF
        i = i + 1
        If i <= maxItemNumber Then
            wSeriesNumber(i) = rs("SeriesNumber")
            wItemName(i) = rs("SpItemName")
            wQuantity(i) = rs("Quantity")
            wSpecialDemand(i) = rs("CookingSpecialDemand")
            wAddTime(i) = rs("AddTime")
            wTableNumber(i) = rs("Table")
            wWaiterNumber(i) = rs("WaiterNumber")
            wChItemName(i) = rs("ItemName")

            wnLabel(i).Caption = rs("SpItemName")
            wqLabel(i).Caption = rs("Quantity")
            If rs("TableNumber") = counterNumber Then
                wspLabel(i).Caption = "TO GO " + rs("CookingSpecialDemand")
            Else
                wspLabel(i).Caption = rs("CookingSpecialDemand")
            End If
            wtLabel(i).Caption = Mid(rs("AddTime"), 1, 5)
            wtnLabel(i).Caption = FormatNumber(rs("AddMoney"), 2, vbTrue)
        End If
        rs.movenext
    Loop
    oConn.Close
    wItemNumberOld = wItemNumber
    wItemNumber = i
    i = i - wItemNumberOld
    If i > 0 Then
        If playAgent Then Peedy.Show
        If i = 1 Then
            Peedy.Speak "Hi, there is a new order!"
        Else
            Peedy.Speak "Hi, there are " + CStr(i) + " new orders!"
        End If
        Peedy.Hide
    End If
End Sub

Private Sub WaitingItemShow()
    For i = 1 To maxItemNumber
        wlTop = wfTitle.Height + (i - 1) * (wlHeight + cInterval)

        wnLabel(i).Top = wlTop
        wnLabel(i).Left = nlLeft
        wnLabel(i).Width = nlWidth - margin / 2
        wnLabel(i).Height = wlHeight
        wnLabel(i).Visible = True

        wqLabel(i).Top = wlTop
        wqLabel(i).Left = qlLeft
        wqLabel(i).Width = qlWidth - margin
```

```
        wqLabel(i).Height = wlHeight
        wqLabel(i).Visible = True

        wspLabel(i).Top = wlTop
        wspLabel(i).Left = splLeft
        wspLabel(i).Width = splWidth - margin
        wspLabel(i).Height = wlHeight
        wspLabel(i).Visible = True

        wtLabel(i).Top = wlTop
        wtLabel(i).Left = tlLeft
        wtLabel(i).Width = tlWidth
        wtLabel(i).Height = wlHeight
        wtLabel(i).Visible = True

        wtnLabel(i).Top = wlTop
        wtnLabel(i).Left = tnlLeft
        wtnLabel(i).Width = tnlWidth
        wtnLabel(i).Height = wlHeight
        wtnLabel(i).Visible = True
    Next i
End Sub

Private Sub Winsock1_Close()
    Call Winsock1.Close
End Sub

Private Sub Winsock1_Connect()
    Call Winsock1.SendData(strMessage)
End Sub

Private Sub Winsock1_Error(ByVal Number As Integer, Description As String, ByVal Scode As Long, ByVal Source As
String, ByVal HelpFile As String, ByVal HelpContext As Long, CancelDisplay As Boolean)
    MsgBox Description
    Winsock1.Close
End Sub

Private Sub wItemUpdate(wIndex As Integer)
    If wnLabel(wIndex).Caption <> "" Then
        'PrintOrder wIndex
        sSQLQuery = "UPDATE OrderItem SET Status=1 WHERE SeriesNumber=" + wSeriesNumber(wIndex)
        oConn.Open dbEngine
        rs.Open sSQLQuery, oConn
        oConn.Close
        ReadWaitingItem
        ReadBeginningItem
    End If
End Sub

Private Sub Winsock1_SendComplete()
    Call Winsock1.Close
End Sub
```

REFERENCES

[1] Chieh-Chou Chou; Project proposal: mOrder-Server Food Service, March, 2002

[2] James Y. Wilson, Aspi Havewala; Building Powerful Platforms with Windows CE, Addison Wesley Professional, 2001

[3] Douglas Boling, Programming Microsoft Windows CE, Microsoft Press, 2001

[4] Douglas Dedo, etc., Mobile Enterprise Solutions: What is the Appropriate Pocket-Size Platform, COMDEX, 2001

[5] Simon Robinson, etc., Professional C# 2nd Edition, Wiley Publishing Inc., 2003

[6] H.M.Deitel, Visual Basic.Net for Experienced Programmers, Pearson Education Inc., 2003