

E-STAURANT: A SOFTWARE INFRASTRUCTURE FOR RESTAURANT
MANAGEMENT

By

RAJANIKANTH KANYABOINA

A THESIS PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

UNIVERSITY OF FLORIDA

2001

Copyright 2001

by

Rajanikanth Kanyaboina

I dedicate this thesis to my family

ACKNOWLEDGMENTS

I offer my highest gratitude to Dr. Abdelsalam Helal for providing me with the guidance and motivation to complete this thesis. I also thank Dr. Paul Fishwick and Dr. Sanguthevar Rajasekaran for serving on my thesis committee. It makes me feel very proud to have people of their stature related to my work. I would also like to thank all my colleagues in the Harris Lab for the day-to-day learning experience they shared with me.

I am thankful to all my friends, for their invaluable help during the course of this thesis.

On a more personal note, I would like to thank my family whose love, support, and constant encouragement were of great importance through this work.

TABLE OF CONTENTS

	<u>page</u>
ACKNOWLEDGMENTS.....	iv
LIST OF FIGURES.....	vii
ABSTRACT	ix
 CHAPTERS	
1 INTRODUCTION.....	1
2 RELATED WORK	4
2.1. Distributed Development Under Windows CE.....	4
2.1.1 Windows CE And Win32.....	5
Kernel	5
GWES.....	6
GWES and windows	6
GWES and UI.....	6
Communications.....	6
2.1.2. Programming Conditions	6
Programming Environment.....	6
Memory Management	7
Power Management.....	8
Operating States	9
Designing a User Interface for an H/PC.....	10
The H/PC Shell.....	10
The Taskbar.....	11
Windows Design	11
Hardware Considerations	12
Communications and Connectivity	12
Windows CE Communications Architecture	12
2.1.3 Comparison Between PalmOS and Windows CE.....	13
2.2. The iPAQ.....	15
2.2.1. Feature Summary	16
2.2.2. Writing on iPAQ	17
2.2.3. Communication with Desktop.....	18
2.3. iPAQ and Wireless LAN.....	25

2.3.1 Working of Wireless LAN	26
2.3.2 Configuration of Wireless LAN	27
2.4 Survey of related solutions	28
3 ARCHITECTURE	30
3.1 Introduction	30
3.2 Architecture	31
3.2.1 Restaurant Server	32
3.2.2 WebClient	33
3.2.3 Chef	37
3.2.4 Waiter	38
3.2.5 Concierge	41
3.2.6 Customer	43
4 IMPLEMENTATION: E-STAURANT	46
4.1 RestServer: The Server side component	46
4.2 RestaurantMenu: Client process on iPAQ	47
4.3 Waiter: Table Service Management	49
4.4 WebClient	50
4.5 Summary	51
5 EXPERIMENTS	52
5.1 Introduction	52
5.2 Conditions and Assumptions	52
5.2.1 Analytical model of a Traditional restaurant	53
5.2.2 Analytical Model of e-Staurant	54
5.3 Simulator	55
5.3.1 Working of the simulator	56
5.4 Experimental Results	58
5.5 Conclusion	59
6 CONCLUSION	60
6.1 Goals Accomplished	60
6.2 Future Work	61
LIST OF REFERENCES	63
BIOGRAPHICAL SKETCH	65

LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
2.1: Windows CE architecture	4
2.2: Windows CE communication architecture.....	13
2.3: Front panel of iPAQ	17
2.4: Initial screen upon inserting the CD.....	19
2.5: Selecting the installation folder for the desktop.....	20
2.6: Getting connected.....	20
2.7: Defining a new partnership for your PC companion.....	21
2.8: Setting up a partnership.....	21
2.9: Setting up a partnership - choosing the number of PCs to sync with.....	22
2.10: Selecting the synchronization settings for a new partnership	22
2.11: New partnership is complete.....	23
2.12: Synchronizing the first time	23
2.13: Mobile devices folder.....	24
2.14: Sync mode	24
2.15: Sync options	24
2.16: Sync rules	25
2.17: Initial screen	27
2.18: Setting up network name.....	27
2.19: Network settings.....	28

3.1: Restaurant communication model.....	30
3.2: Communication model of the restaurant server	32
3.3: Communication model of webclient	34
3.4: Welcome screen of the webclient.....	35
3.5: Authentication of manager login	36
3.6: Screen to add a item to the menu	36
3.7: Communication model of the chef.....	37
3.8: Interface provided to the chef to indicate status of the item.	38
3.9: Communication model of the waiter.....	39
3.10: Screen showing the status of the tables whose items are cooked.....	40
3.11: Communication model of the concierge	41
3.12: Finding out table for the party arrived.....	42
3.13: Interface provided to the concierge to enter the customer information.....	42
3.14: Communication model of customer	43
3.15: Welcome screen	44
3.16: Choosing the type of menu.....	44
3.17: Categories of menu	45
3.18: Images indicating type of the items.....	45
3.19: Selection of spice	45
3.20: Images indicating waiting time	45
5.1: Representation of the traditional model	53
5.2: Representation of e-Staurant.....	54
5.3: Various operations of the simulator	56
5.4: Delay Analysis	58
5.5: Comparison plot for Load sharing.	59

Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Science

E-STAURANT – A SOFTWARE INFRASTRUCTURE FOR RESTAURANT
MANAGEMENT

By

Rajanikanth Kanyaboina

August 2001

Chairman: Dr. Abdelsalam (Sumi) Helal

Major Department: Computer and Information Science and Engineering

Advances in wireless networking technology have engendered a new paradigm of computing called mobile computing, in which users carry portable devices having access to a shared infrastructure bringing an end to the tyranny of geography. The availability of ultra-portable mobile computing hardware ushers in the era of "any time any where computing" empowering the user with unlimited access in all situations. The use of computing and communications devices becomes an integral part of modern lifestyle, augmenting physical reality with computer-generated information.

Providing an optimal support for the user based on mobile computing hardware poses new challenges for developers of interactive graphics and multimedia information systems. In a novel attempt to explore fully and stretch the unlimited potential that the resources of mobile computing has at its disposal, this thesis proposes an e-business model to incorporate the glow of mobile computing into a more pragmatic piece of

software which targets restaurants extending valuable and more gratifying service by reducing customer waiting time.

e-Staurant, an unparagoned, powerful tool, will help coordinate and streamline every facet of a restaurant management operation to get more customers seated, served and satisfied. It will not only please customers arriving, it will be able to please a greater number of them. e-Staurant helps to turn more tables, more often, which means turning higher profits. From the customers point of view it not only reduces their waiting time but also utilizes their time.

The design, architecture and implementation of e-Staurant are presented. An analytic queuing model is derived to quantify the impact of e-Staurant on waiting time (pleasing customers) and throughput (pleasing restaurant manager).

CHAPTER 1 INTRODUCTION

Recent years have witnessed the rapid growth of mobile computing environments. The defining characteristic of mobile computing is the attempt to break away from the traditional desktop computing paradigm and move computational power into the environment that surrounds the user. The handheld device is seen more and more as the most effective solution to mobile working practice. The emergence of powerful portable computers, along with the advances in wireless technologies, has made mobile computing a reality. e-Staurant is an attempt to exploit the unlimited potential that the resources of mobile computing has at its disposal.

1.1 Goal of Thesis

This thesis targets to provide a software infrastructure to the restaurant management automation. Each segment of automation addresses a portion of the process of seating the customers, pleasing the customers and increasing number of the customers. The major restaurant automation opportunities provided are

- **Table Management.** When customer arrives it looks at the current state of the restaurant and finds out the perfect match according to the preferences of the customer.
- **Waiting.** If seats are not available it indicates the approximate amount of waiting time to the customer and at the same time utilizes his waiting time by allowing him to browse through the interactive menu provided on the iPAQ and order.

- **Menu.** Pleases the customer by providing a very good interactive menu and detailed description of the food items.
- **Checkout.** Provides an efficient way to the customer to do check out or submit his order in an efficient and less time-consuming way.
- **Web Interface.** It allows the manager of the restaurant to change the menu of the day and to allot work among the chefs of the restaurant.
- **Chef Interface.** Provides a user interface to the chefs to give an idea of the items to be cooked and to send a confirmation to the Waitress that a particular order is ready.
- **Waiter Interface.** Provides a user interface to the waitress to indicate that particular customers items are ready.

As minutes are eliminated from each segment, cumulative effect is dramatic. If you enter a restaurant on a busy day the major amount of time is wasted in getting seated and reached by the waitress, e-Staurant emphasizes and utilizes this unwanted portion of the customer's valuable time. In e-Staurant model when customer first arrives at the restaurant, he is given handheld device by taking his financial details and if seats are available he will be directed to appropriate seat or he will be requested to wait for some time, but from that point he will be allowed to view the menu and can go ahead and order. By the time he reaches and settles down at his table, his order will be processed in the kitchen, and the waitress will be reaching the table with food.

One more notable feature that e-Staurant exploits in restaurant management is checkout of a customer. In a normal restaurant once you are done with your eating you have to wait for the waiter to give you bill, have your credit card transaction approved

and get it back to you. In this model once you are done you can commit your order on the handheld device provided to you and view your bill and approve it and get a confirmation immediately. These are features it provides from the customer point of view. From management point of view it tries to reduce the time taken by the concierge to allot the table for arrived customer and always keeps waitresses notified about the status of the customers as well as the chefs, Thereby increasing the throughput of the restaurant.

It efficiently manages all the transactions taking place in the restaurant. From customer point of view it mainly targets reduction of unwanted waiting time and improvement of the service that can be provided to him by proper utilization of his time. It helps restaurant management by increasing the throughput of the system and automates all the transactions in the restaurant.

Chapters 2 discuss the background of distributed development under Windows CE environment and also provides support for enabling iPAQ wireless . In Chapters 3 and 4 we fully outline the detailed description and implementation of e-Staurant. Finally, Chapter 5 concludes with a quantitative analysis of the performance of e-Staurant and discussion of future research in the area of handheld devices.

CHAPTER 2 RELATED WORK

e-Staurant is a powerful tool targeted to optimize every operation of Restaurant management. The significant part of it constitutes development under Windows CE environment for iPAQ. This chapter provides a detailed description of the Windows CE environment and outlines its advantages over the PalmOS. A detailed description of the iPAQ and its configuration to the wireless environment is also provided. Finally chapter concludes with a survey of concepts and technologies that are similar to e-Staurant

2.1. Distributed Development Under Windows CE

The Windows CE operating system is a 32-bit, multitasking, multi-threaded operating system that has an open architecture design, providing support for a variety of devices. Windows CE also has the advantage of being portable, providing choice in microprocessors and has integrated power management, enabling long battery life on mobile devices [1]. Windows CE is an OS based on the Win32 API.

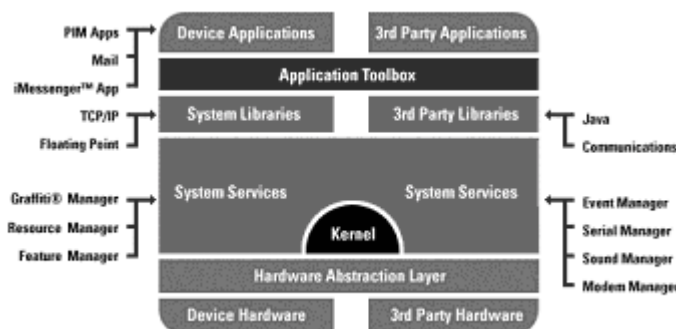


Figure 2.1: Windows CE architecture¹

¹ This architecture is taken from the Microsoft web site.

When deciding a Windows CE-based program, first factor to be determined is the configuration of the hardware platform and shell for which they are developing. Because Windows CE is a modular operating system, an original equipment manufacturer (OEM) chooses specific modules and components of Windows CE to configure Windows CE-based devices in many different ways. Within the H/PC device category itself, OEMs may choose to implement different components and hardware options from one another.

Some critical modules of the WCE operating system: the kernel, the object store, the graphics, windowing and events subsystem (GWES), and communications. Windows CE also contains additional, optional modules that support such tasks as managing installable device drivers and supporting COM.

2.1.1 Windows CE And Win32

Windows CE was designed separately from Windows NT and Windows 95, yet is similarly based on the Windows 32-bit API programming model. The Windows CE modules are broadly classified as [2]

1. Kernel
2. Object Store
3. GWES
4. Communications

Kernel

The kernel is the core of the OS and is represented by the coredll module.

- The kernel provides the base operating system functionality.
- The kernel is responsible for memory management, process management, and various file management functions.
- The kernel manages virtual memory, scheduling, multitasking, multithreading and exception handling.

GWES

GWES is the graphical user interface between a user, application written, and the WCE OS. GWES handles user input by translating user input into messages that convey information to applications and the WCE OS.

GWES and windows

The window is the central element of the GWES module. Win32 applications that are built to run on WCE device can use windows to receive messages from the OS- whether or not the application has a user interface.

GWES and UI

GWES provides Auto PC Standard Controls and other resources for applications that implement a user interface (UI).

Communications

The communications module in WCE enables the device to exchange information with a variety of devices using a number of different communications technologies.

2.1.2. Programming Conditions

The following list highlights important areas to consider when programming and H/PC application [3]:

- Programming environment
- Memory management
- Power management
- User interface design
- Communications and connectivity

Programming Environment

Microsoft embedded visual C++ is chosen as environment. To debug, test and develop the code, an emulator provided by Windows CE IDE has been used.

Memory Management

Low-memory situations are more common on an H/PC than on a desktop computer because the compact size of an H/PC limits the number of memory chips that can be installed. Windows CE circumvents this obstacle by using the WM_HIBERNATE message as its primary mechanism for requesting applications, to free memory. When freeing memory is required, the system posts the message to one or more applications, beginning with the application that has been inactive the longest.

Windows CE uses a series of memory thresholds to conserve system resources. Table shows values for memory thresholds. The values are based on a 1-KB memory page; values for a 4-KB memory page are provided in parentheses.

Hibernating is vital for applications running on an H/PC. When it receives a WM_HIBERNATE message an application should be programmed to:

- Free any large blocks of memory that were allocated by VirtualAlloc—for example, a cache.
- Free as many Graphics, Windowing, and Events Subsystem (GWES) objects as possible, including windows, bitmaps, and device contexts, because they use large amounts of memory.
- Save the state of the heap in order to restore it later, and then free the entire heap.

When the system brings an application to the foreground, it sends a WM_ACTIVATE message. Applications should be programmed with a WM_ACTIVATE handler that can restore memory resources following hibernation.

Table 2.1: Different memory thresholds for H/PC

Threshold	Value	Description
Hibernation threshold	128 KB (160 KB)	The point at which a system enters a limited-memory state. A system sends a WM_HIBERNATE message when its memory falls below this value.
Low-memory threshold	64 KB (48 KB)	The minimum available memory that a system must maintain when it is in a low-memory state.
Critical-memory threshold	16 KB (48 KB)	The minimum available memory that a system must maintain when it is in a critical-memory state.

Power Management

The primary function of power management is to increase the battery life of an H/PC. Power management is accomplished by providing accurate estimates of remaining battery life and notifying a user when batteries are nearly depleted. Power management for an H/PC is based on the following assumptions:

- The H/PC is used less than two hours per day in bursts from five minutes to one hour.
- The display is powered 100 percent of the time during use.
- The device runs less than 10 percent of the time during typical use.
- The device uses both main batteries and a backup battery.
- User data is stored in RAM (volatile memory), so provisions for backing up data should be provided.
- PC Cards and other storage devices that draw appreciable power from internal batteries significantly reduce battery life.

Operating States

An H/PC manages power by allowing the operating system to automatically select one of these operating states, based on user activities and programming activities. These states are dead, suspend, and on.

In the dead state, an H/PC uses no power. It has no batteries, or the batteries have no power. All contents in RAM are lost. The user purchases a device in this state.

In the suspend state, an H/PC uses minimal power to maintain its clock, its applications, and the persistent data stored in RAM. To reduce power requirements, an H/PC removes power from unneeded circuits and devices, such as the keyboard decoder, display, scratch pad memory, and processor. A PC card storage device driver determines the power that it uses when the device is in the suspend state. The processor might take as long as 100 milliseconds to wake up from this state.

An H/PC switches to the suspend state when the following events occur:

- The user selects the Suspend command.
- The device detects a critical-power condition.
- The activity timer performs a time-out.

An H/PC is in the on state any time the display, keyboard, or touch screen is active. When in the on state an H/PC can switch between two processor modes: full speed and idle. In full-speed mode, the processor runs at normal operating frequency. In idle mode, the internal processor clock stops, and the processor uses little power.

Peripherals and dynamic RAM (DRAM) may be on or off. The processor can enter and exit this state in approximately 10 milliseconds. An H/PC switches to the on state when one of the following events takes place:

- The user presses the On button.
- The user triggers an alarm event.
- The user performs a warm or cold boot.
- The user changes the battery.

In the on state, the default mode is full speed. An H/PC switches from full-speed mode to idle mode when all processes are idle. Switching to idle mode is transparent, both to the user and to most applications, because the system continues to process interrupts, including the time-slice interrupt.

Designing a User Interface for an H/PC

One of the key design goals behind the H/PC is lowering the barriers to entry for new users. The user interface for Windows CE is designed to take advantage of the experienced user's comfort with the Windows 95 interface. Users of Windows CE will find that accessing documents, launching applications, switching among tasks, browsing the file hierarchy in Windows CE, and other common activities are as familiar and easy to perform as they are in Windows 95.

On an H/PC, the desktop forms a visual background for all operations. It provides a familiar interface for accessing documents, launching applications, switching between tasks, browsing the file system, and performing other services. The parts of an H/PC desktop include a work area, a taskbar, and application shortcuts or icons, such as the Recycle Bin and Inbox.

The H/PC Shell

The shell is similar to other Microsoft Windows desktops. It contains file, folder, and shortcut icons that a user can position anywhere on the desktop. However, unlike other Windows-based platforms, an H/PC has a virtual border around to prevent icons from being fully obscured by the screen edge or taskbar. The desktop does not permit a user to position an icon beyond the boundaries defined by the following rectangle coordinates:

(0,0, CX-16, CY-16)

where CX and CY are the width and height of the screen.

The Taskbar

A taskbar is used to switch between or minimize open windows and to access global commands and other frequently used objects. The taskbar contains a start button, window buttons, and a status area. It also contains a desktop button that provides quick access to the desktop from any application. Because H/PC applications do not have title bars, users identify a running application primarily by the icon and text displayed on its taskbar button.

By default, the taskbar is the top window in the shell. When fully displayed, the taskbar is 26 pixels high and either 480 or 640 pixels wide, depending on the resolution of the display. In the shell, the taskbar takes on added flexibility. When hidden, the taskbar is 5 pixels tall. When a user hides the taskbar, a notification is sent to all applications that the usable vertical screen is increased by 21 pixels. To reactivate a hidden taskbar, each H/PC touch screen contains a 2.5 mm tap region along all four edges of the display. This, combined with the height of the taskbar, provides a generous tap region for activating the taskbar.

Windows Design

The design of windows is another important element in creating a UI for the H/PC. Windows enable a user to view and interact with data. Consistency in window design is important because it enables users to easily transfer their skills and focus on tasks, rather than learn new conventions.

Primary windows in Windows CE for the H/PC are similar to windows found on desktop-based applications. H/PC applications can use all of the standard menus and controls available in Windows CE.

Hardware Considerations

While all H/PCs conform to a hardware specification standard, some OEMs offer features that extend functionality. These additional features include an expanded LCD screen, 640 x 240 or 640 x 480 pixels in size. The expanded screen allows applications to display up to 80 character per line at an easily readable character size, using half-height of full-size VGA graphics resolution. This means the font aspect ratio is similar to a desktop computer and eliminates the feeling that the characters are squeezed onto the screen.

Communications and Connectivity

One of the strengths of the H/PC is its ability to connect to and communicate with other computers and peripherals.

Windows CE Communications Architecture

One of the key features of Windows CE-based devices is the ability to communicate with other devices [4]. Windows CE supports two basic types of communication: serial communication and communication over a network. Most devices feature built-in communications hardware, such as a serial port or an IR transceiver. The NDIS implementation on Windows CE supports the following communications media: Ethernet (802.3), Token Ring (802.5), IrDA, and WAN. Diagram 2.2 outlines the communications architecture of the Windows CE operating system.

In the Windows CE communications architecture, the NDIS interface is located below the IrDA, TCP/IP, and point-to point protocol (PPP) drivers. The NDIS wrapper

presents an interface to the upper and lower edges of a miniport driver. To an upper-level driver, such as the TCP/IP protocol driver, the NDIS interface looks like a miniport driver. To the miniport, the NDIS interface looks like an upper-level protocol driver. On the bottom of the communications architecture, the NDIS interface functions as a network adapter driver that interfaces directly with the network adapter at the lower edge. At the upper edge, the network adapter driver presents an interface to allow upper layers to send packets on the network, handle interrupts, reset or halt the network adapter, and query or set the operational characteristics of the driver.

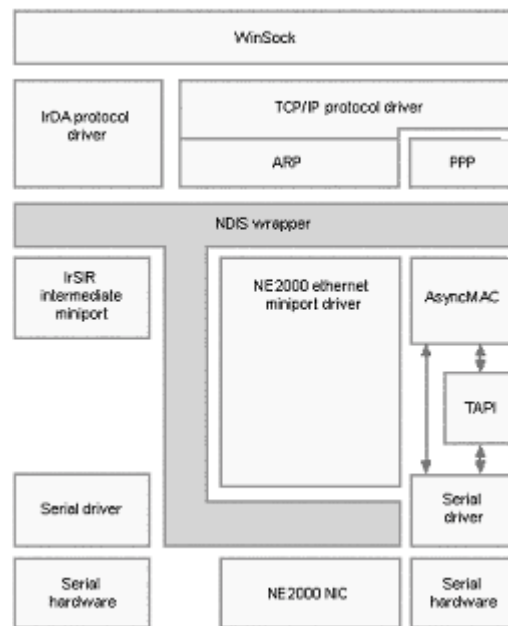


Figure 2.2: Windows CE communication architecture²

2.1.3 Comparison Between PalmOS and Windows CE

The iPAQ was designed to be a handheld computer complete with multimedia functionality, while Palm was designed to be PDA³ with wireless connectivity. This

² This diagram is taken from the Microsoft web site

³ PDA is Personal Digital Assistant

makes it very difficult to compare two architectures but a general overview of their strengths is discussed here [5].

PalmOS is single tasking and designed to support only the hardware base implemented by Palm in its devices. Windows CE is a multitasking environment designed to run on multiple hardware bases and processors, therefore it has much high overhead in RAM, physical memory and processor use. The iPAQ H3650 comes with 16MB of flash-ROM and 32MB of SDRAM; supporting DRAM, ROM, flash-ROM, and SRAM and virtual memory, with up to 96MB of physical main memory. The Palm, has no caches, no virtual memory, and supports a maximum of 12MB of RAM. The iPAQ has a typical battery life of 12 hours, using a lithium-ion rechargeable battery; its processor consumes just less than 400mW while in operating mode. The Palm⁴ has a battery life of two to four weeks using two AAA batteries, its processor consumes on average 54mW while operating.

Palm chose Motorola's Dragonball VZ processor to provide a simple interface and operating system for their Palm Pilots. Their was to "provide instant access to critical schedules, addresses, memos, to do lists and fit in a shirt pocket. For iPAQ Compaq decided upon Intel's StrongARM processor. Their intent was to provide "high performance, robust functionality and versatility while meeting the small size and low power restrictions of portable, battery operated products". The StrongARM CPU is designed to have all the functionality of desktop processor, while the Motorola Dragonball is designed to address the limited needs of a handheld device. The StrongARM processor is a RISC processor. RISC has a faster instruction completion

⁴ Here Palm refers to Palm VIIx

times due to its use of many primitive instructions to complete each instruction, where as CISC (Palm Processor) allows for more complex instructions to be run at slower speeds.

PalmOS is extraordinarily simple compared to Windows CE. PalmOS is specialized for defined hardware platform, where as Windows CE must be able to be used on multiple hardware bases. This means that overhead for palmOS is much less than WindowsCE, which must be generalized enough on variety of processors and use a variety of components. PalmOS is a single tasking environment, meaning it does not allow programs to run in back ground. WindowsCE is multitasking which is good for the end user, but results in even more overhead in the operating system, as it now requires virtual memory and memory protection in order to allow speedy task switching and keep tasks conflicting from one another.

2.2. The iPAQ

iPAQ is a multimedia-centric PDA with versatile expansion capabilities from Compaq. The most appealing feature of the iPAQ is its bright, crisp color screen. Its unique hardware features include a headphone jack for listening to digital music (or even just the built-in alert sounds), an infrared port for beaming data to other similarly equipped Pocket PCs, and a voice recorder for capturing audio. The device is powered by a rechargeable lithium-ion battery that performed acceptably for a color display.

In terms of storage and software, the iPAQ is a modern workhorse. It has 32 MB of RAM for storing large media files, plus 16 MB of ROM that houses the included applications. These include the Microsoft Windows CE 3.0 operating system, Pocket PC versions of Microsoft Word, Excel, Internet Explorer, Money, Windows Media Player, and Reader as well as applications like a calendar, contact list, task list, and notes.

Another notable is the iPAQ's Navigator, a large four-direction button below the screen that can be used to scroll through selections. Looking at the Qstart screen (an icon view of most of the programs), different icons can be highlighted using the Navigator.

However, counter intuitively, the Navigator has to be pushed up and down to highlight objects to the left and right, respectively. Pushing the center of the Navigator acts like the Enter key on a computer keyboard, and allows to select a program or file. Compaq iPAQ.

2.2.1. Feature Summary

Processor:	206-Mhz Intel StrongARM SA-1110 32-bit RISC Processor.
Memory:	64-MB or 32-MB SDRAM depending on model, 16-MB Flash ROM Memory
Power Supply:	950 mAh Lithium Polymer, rechargeable in docking cradle or with AC.
Operating System:	Windows powered Pocket PC
RAM:	32 MB/64 MB
ROM (Flash):	16 MB
TFT Color Display:	Number of Colors: 4096 Resolution (WxH): 240 x 320 Screen size (dimensions): 2.26" x 3.02"
Connections:	Infrared port Standard serial port USB port
Weight:	6.3 oz

Size (dimensions): 5.11" x 3.28" x 0.62"

All the features are taken from the compaq's manual [6].

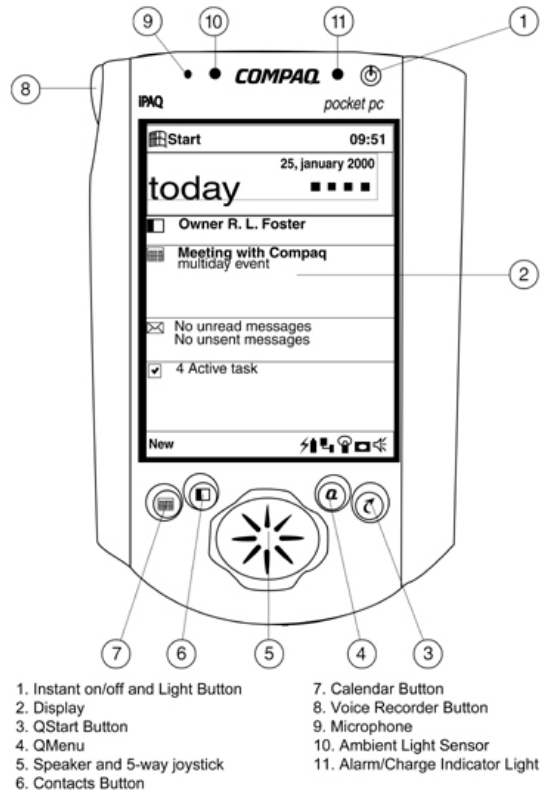


Figure 2.3: Front panel of iPAQ⁵

2.2.2. Writing on iPAQ

There are three different ways to enter information [7].

1. Typing the information on desktop and then downloading it to iPAQ (using ActiveSync)
2. Tapping virtual keyboard on the iPAQ's screen.

⁵ This figure is taken from Compaq's web site

3. Using the handwriting-recognition option (where letters are to be written onto the PDA screen using the stylus.)

2.2.3. Communication with Desktop

The iPAQ interfaces with Windows-based PCs using Microsoft's ActiveSync 3.1 software [8]. The synchronization process is pretty efficient, and allows to browse the contents of the iPAQ from desktop. All PDAs upload and download information through a process called syncing, short for synchronization. The process is fairly simple, once it is set up. A copy of all the information needed on both machines can be created by plugging a serial or USB cradle into Desktop. Palm OS-based devices call this process as HotSync, PocketPCs call the same process ActiveSync. By default, the syncing process will back up date book, address book, to-do lists, and anything else that is to be on PDA. The syncing process also enables download of different executables and software programs on to the device. ActiveSync synchronizes only the files that are kept in special folder and designated to synchronize it wont synchronize all the files present on the device. If a update is done on a synchronized file, the next time when device connects to the desktop, active sync copies the file to the desktop or replace any older versions there. If device and desktop are connected synchronization of files is done immediately.

Installing ActiveSync:

ActiveSync installation procedures are outlined below.

- To support an ActiveSync connection using USB, Host PC must be running Windows 98 or Windows 2000. Microsoft does not support an ActiveSync USB connection under Windows 95 or Windows NT.
- ActiveSync version 3.1 must be installed on host PC before the iPAQ is connected for the first time. If the iPAQ is connected to the PC first, it

will not be recognized properly, and will be listed as an unknown USB device. If ActiveSync is installed first, when the iPAQ is connected for the first time it will be recognized properly, and will be listed under the "Windows CE USB Devices" heading in Device Manager.

- If the iPAQ was connected to the PC before ActiveSync was installed, the iPAQ's USB device record in Device Manager (Go to Settings -> Control Panel -> Add/Remove Programs) must be uninstalled. After the device is removed, and ActiveSync is installed, the iPAQ must be reconnected. It will be recognized properly and listed under the "Windows CE USBDevices" heading.

A quick look at the installation procedure.

Upon inserting the CD the option to install ActiveSync 3.1 is seen.

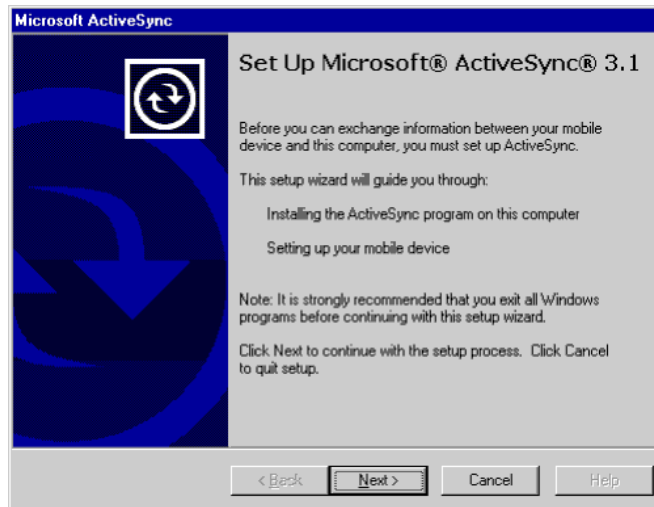


Figure 2.4: Initial screen upon inserting the CD

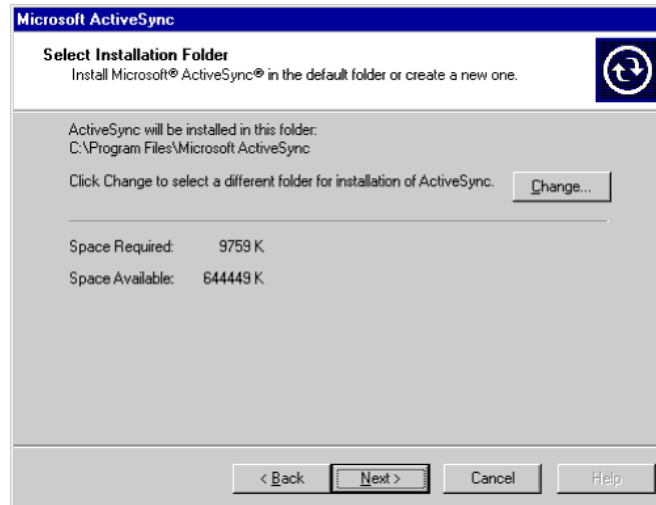


Figure 2.5: Selecting the installation folder for the desktop



Figure 2.6: Getting connected

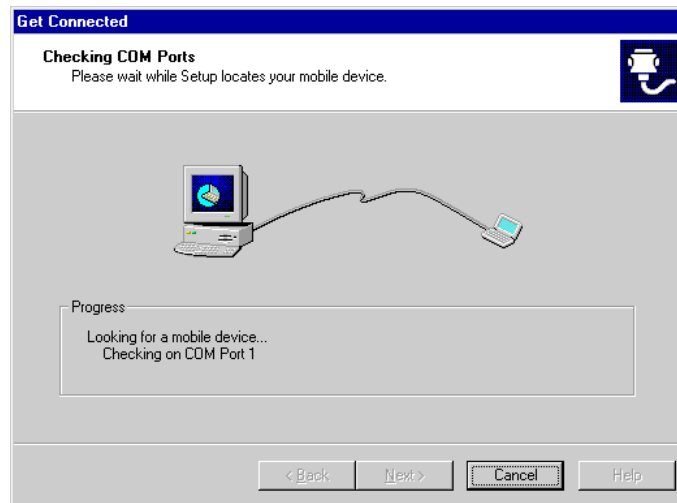


Figure 2.7: Defining a new partnership for your PC companion



Figure 2.8: Setting up a partnership

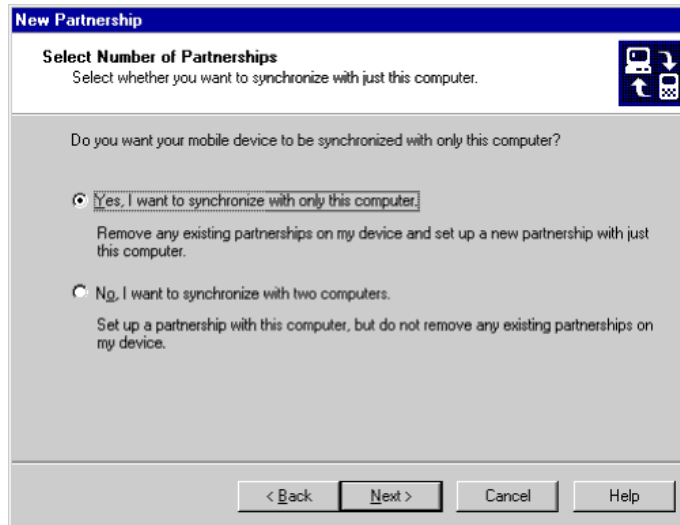


Figure 2.9: Setting up a partnership - Choosing the number of PCs to sync with

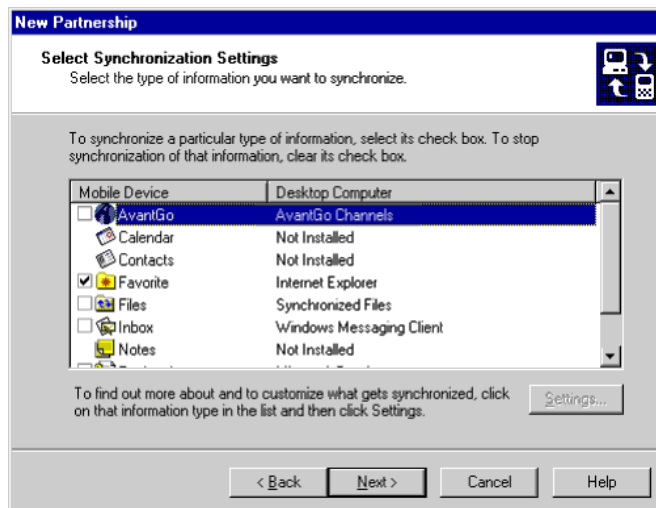


Figure 2.10: Selecting the synchronization settings for a new partnership

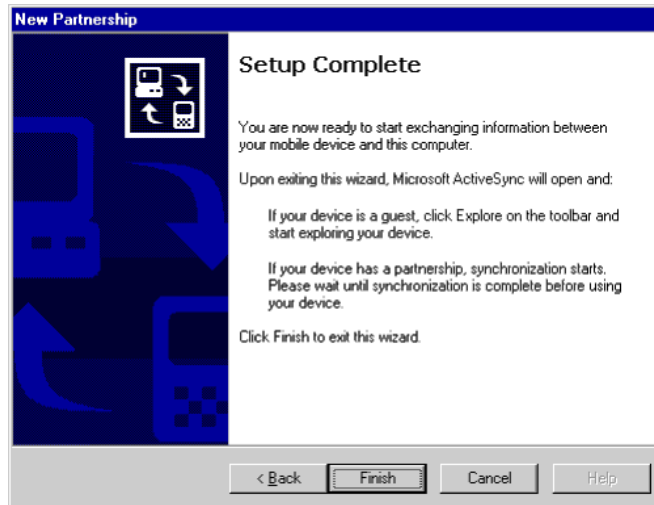


Figure 2.11: New partnership is complete

Synchronizing: ActiveSync 3.1 installs its own PPP server for synchronization. It also offers advanced capabilities like auto-detection of the serial port of the PC companion (desktop) installed on. It also automatically adjusts the baud rate of the connection-based settings of the PC companion. If ActiveSync 3.1 runs into a problem completing synchronization, it will notify in the status window and an option is also given to see the log and to identify the problem and potential options to resolve it.

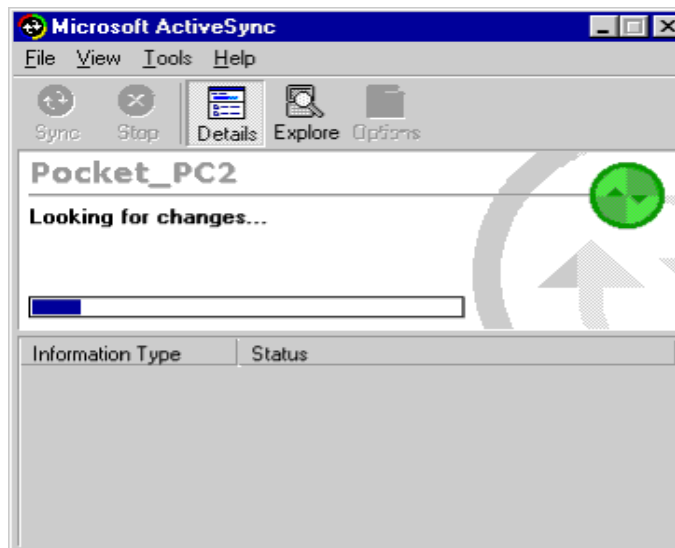


Figure 2.12: Synchronizing the first time



Figure 2.13: Mobile devices folder

Configuring ActiveSync 3.1

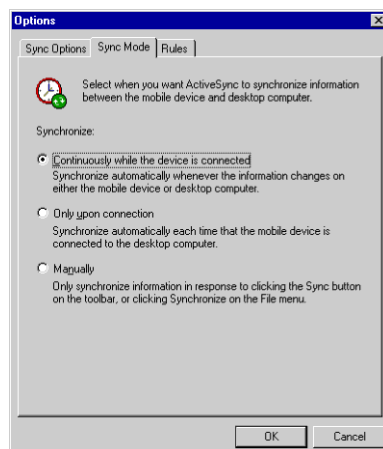


Figure 2.14: Sync mode

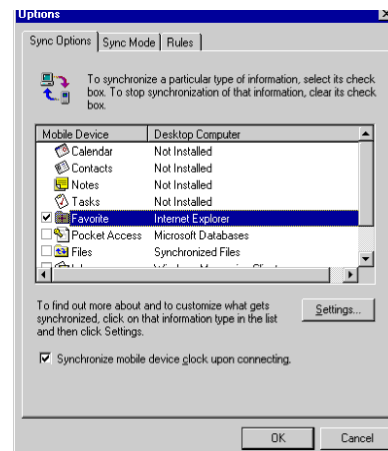


Figure 2.15: Sync options

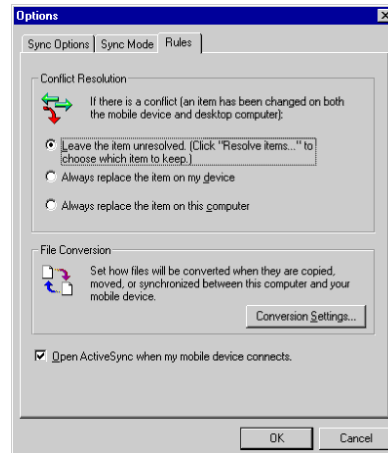


Figure 2.16: Sync rules

Problems Encountered: One of the serious problem that took lot of our time is "Application CEMGRC.EXE has performed an illegal operation and will shut down. Exception: 0xc000001d Address: 0001500c Fatal Application Error." We have seen that problem twice, but it has not been persistent. There is nothing wrong in the device also we ended up checking some things like, not connected, previous copy of application being downloaded already active, low power on iPAQ. And once we ended up doing hard reset.

2.3. iPAQ and Wireless LAN

A wireless local area network (LAN) is a flexible data communications system implemented as an extension to, or as an alternative for, a wired LAN. Using radio frequency (RF) technology, wireless LANs transmit and receive data over the air, minimizing the need for wired connections. Thus, wireless LANs combine data connectivity with user mobility. Wireless LANs use electromagnetic airwaves to communicate information from one point to another without relying on any physical connection.

2.3.1 Working of Wireless LAN

Wireless LANs use electromagnetic airwaves (radio or infrared) to communicate information from one point to another without relying on any physical connection. Radio waves are often referred to as radio carriers because they simply perform the function of delivering energy to a remote receiver. The data being transmitted is superimposed on the radio carrier so that it can be accurately extracted at the receiving end. This is generally referred to as modulation of the carrier by the information being transmitted. Once data is superimposed (modulated) onto the radio carrier, the radio signal occupies more than a single frequency, since the frequency or bit rate of the modulating information adds to the carrier.

Multiple radio carriers can exist in the same space at the same time without interfering with each other if the radio waves are transmitted on different radio frequencies. To extract data, a radio receiver tunes in one radio frequency while rejecting all other frequencies.

In a typical wireless LAN configuration, a transmitter/receiver (transceiver) device, called an access point, connects to the wired network from a fixed location using standard cabling. At a minimum, the access point receives, buffers, and transmits data between the wireless LAN and the wired network infrastructure. A single access point can support a small group of users and can function within a range of less than one hundred to several hundred feet. The access point (or the antenna attached to the access point) is usually mounted high but may be mounted essentially anywhere that is practical as long as the desired radio coverage is obtained.

End users access the wireless LAN through wireless-LAN adapters, which are implemented as PC cards in notebook or palmtop computers, as cards in desktop computers, or integrated within hand-held computers. Wireless LAN adapters provide an interface between the client network operating system (NOS) and the airwaves via an antenna. The nature of the wireless connection is transparent to the NOS.

2.3.2 Configuration of Wireless LAN

For connecting iPAQ to wireless LAN first step is to choose appropriate WaveLan card and its corresponding firmware and drivers. After installing the drivers and synching them to the device, configuration has to be done in the following way.



Figure 2.17: Initial Screen

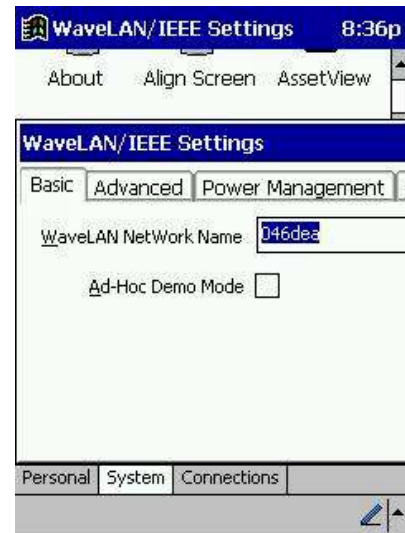


Figure 2.18: Setting up network name

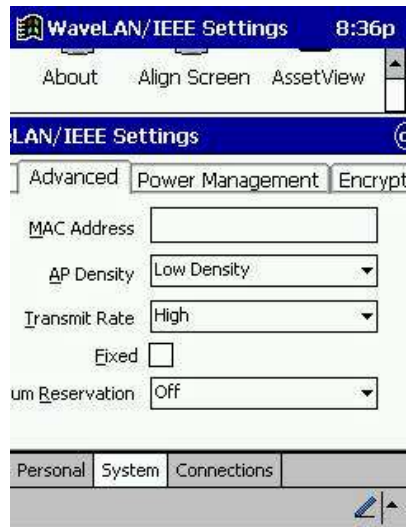


Figure 2.19: Network settings

2.4 Survey of Related Solutions

Advances in wireless networking technology have engendered a new paradigm of computing, called mobile computing, in which users carrying portable devices have access to a shared infrastructure independent of their physical location [9]. This concept of mobile computing has provided the base for e-Staurant. “A Survey of mobile computing technologies and applications “[10] gives a list of references of some non-academic institutions utilizing the concept of mobile computing to provide solutions related everyday life of man. Some of the limiting features of palmtop computers networking capabilities are addressed in BlueSky [11]. “Adding some smartness to devices and everyday things “[12] gives the concept of using the idea of mobile computing to add smartness to everyday things in life.

The following is a commercial solution for restaurant management that is currently available and being used

Pro-host

JCR systems prohost [13] is a table management service. Prohost targets only table management and reservation of tables for the restaurant where as e-Staurant not only targets table management but also deals with the automation chef scheduling, interface for chefs and waiters, remote management for restaurant and providing an interactive electronic menu for the customers.

CHAPTER 3 ARCHITECTURE

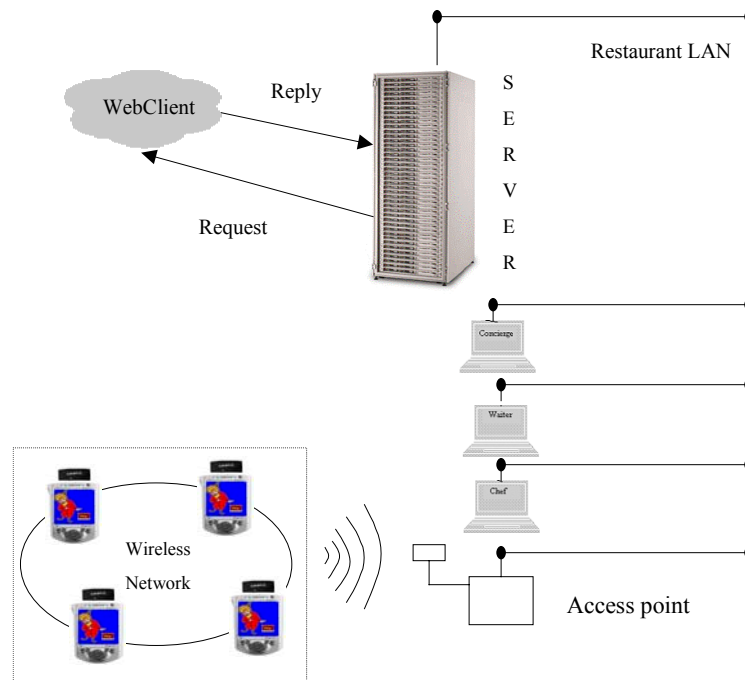


Figure 3.1: Restaurant communication model.

3.1 Introduction

e-Staurant model is implemented with the following view of restaurant.

When customer enters the Restaurant he is greeted by the concierge and the concierge takes his details and checks through the table management service provided by e-Staurant and finds out the exact accommodations for the new customer. Once the table availability is decided, the customer is given a handheld device (which contains the menu), so that he need not wait for the waitress to get the menu. The customer can go

ahead and browse through the menu and order the food items. Centralized Restaurant server will take orders from different customers and then allots them to different chefs and continuously monitors their state to find out whether they are cooked by the chef. Chefs are provided a GUI program, which indicates them the items to be cooked and some information about any special care to be taken while cooking the item. They are also provided with an option to indicate the server that particular item has been cooked. Server takes messages from different chefs and finds out whether all the items ordered by the customer are completed or not. Once they are completed, it indicates the waiters waiting in the kitchen that the order has been completed. Waiter will pick up the order from chefs and goes to the customer's table directly with food. In the time when the customer is placing the order, waiter services can be utilized to serve other customers.

3.2 Architecture

Architecture is targeted to benefit both customers as well the management. From customer's viewpoint the factors considered are

- Total time spent in the restaurant.
- Waiting time for the order.
- Utilization of the waiting time.
- Interactive menu.

From management viewpoint, the factors considered are

- Throughput (total number of customers serviced in a given time).
- Total number of servers required to service a given number of customers.
- Efficient way of managing transactions.
- Global view of the status of the restaurant.

Architecture can be broadly classified as

1. Restaurant server
2. Web client
3. Chef
4. Waiter
5. Customer
6. Concierge

3.2.1 Restaurant Server

A detailed description of operations in the server is shown in Figure 3.2.

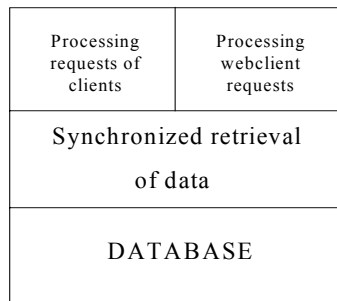


Figure 3.2: Communication model of the restaurant server

Server is designed to provide the following functionality.

- Processes requests from all the clients.
- Run the servlets to provide dynamic data to the web client.
- Synchronize the operations on the database by different clients.

Based on the functionality to be provided server is divided into the following parts.

1. Servlets: To process information for the web clients.

2. Database Component: To provide synchronization of the operations on database.
3. Transaction manager: To process requests by various clients.
4. Centralized database: To store all the transactions in the restaurant.

Transaction manager runs a server program written in Java to listen for all connections from the clients on a particular port, then processes their request and accordingly retrieves data from the database. Synchronization of all these operations is taken care of by an interface called database component. Database component acts like a filter between transaction manager and database. Servlets are run by the Java web server to provide the data to web client.

Technical challenges encountered include building a good interface among different clients. As one of the clients is written in VC++ and remaining in Java, some processing has to be done to change the byte ordering. Also, character conversion has to be done because one of the clients is on iPAQ (Handheld device) and the server is on the desktop computer. A detailed discussion of these issues is provided in Chapter 4.

3.2.2 WebClient

This is the interface provided to the manager of the restaurant to view the status of the restaurant from anywhere.

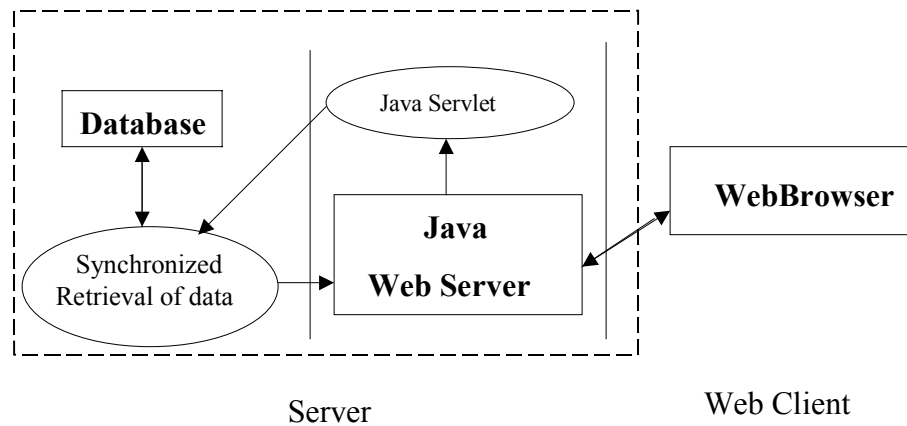


Figure 3.3: Communication model of webclient

This allows the manager of the restaurant to change the menu of the restaurant and allot work to the chefs from any place outside the restaurant. A detailed list of the functions of the webclient is indicated in figure 3. Webclient is an interface to process the remote requests and to direct them to the server. Server with the help of servelets reads the requests of the client and retrieves the data from the Database and sends it back to the client. All the operations on the Database are done through an interface, which synchronizes the operations on the database. Database operations are synchronized to prevent the problems of stale data, as more than one client accesses database at the same time.

The webclient interface allows the manager of the restaurant to modify the data upon authentication as follows.

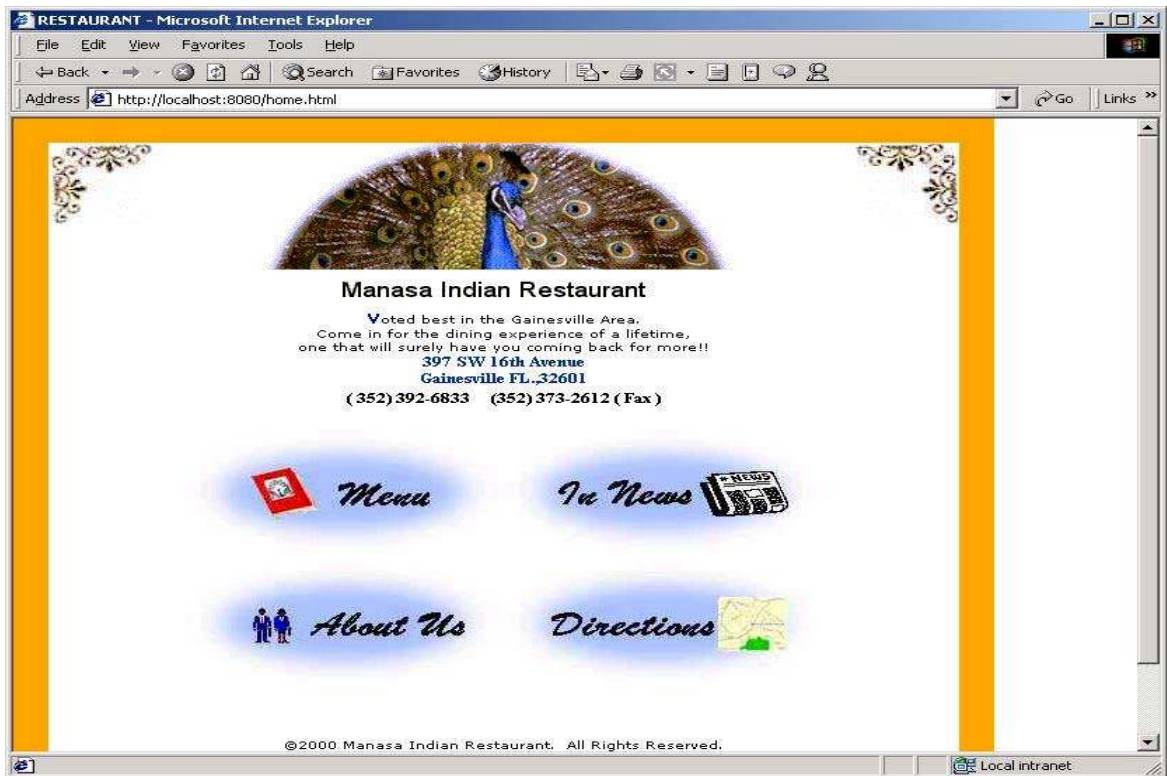


Figure 3.4: Welcome screen of the webclient

The screenshot shows a web browser window titled "RESTAURANT - MENU - Microsoft Internet Explorer". The address bar displays "http://localhost:8080/AdminLogin.html". The page content is framed by a thick orange border. At the top left, it says "Manasa Indian Restaurant". In the center, the heading "Admin Login" is displayed. Below this, there are two input fields: "User Name" and "Password". Under the "Password" field are two buttons: "Enter" and "Cancel". At the bottom of the page, there is a navigation bar with links: "[Home | Menu | InNews | AboutUs | Directions]". Below the links, it says "Manasa Indian Restaurant. All Rights Reserved."

Figure 3.5: Authentication of the manager login

The screenshot shows a web browser window titled "RESTAURANT - MENU - Microsoft Internet Explorer". The address bar displays "http://localhost:8080/servlet/additem?catName=Appetizer". The page content is framed by a thick orange border. At the top left, it says "Manasa Indian Restaurant". In the center, there is a decorative "Menu" header. Below this, the heading "AddItem" is displayed. The form contains several input fields: "ItemName", "Description", "Price", and "Image". Below these is a dropdown menu for "ChefName" with "Prasad" selected. At the bottom of the form are two buttons: "ADD" and "CANCEL". The status bar at the bottom shows "Done" and "Local intranet".

Figure 3.6: Screen to add a item to the menu

3.2.3 Chef

This is a graphical user interface provided to the chef working in the kitchen. A detailed description of its operation is shown in figure 7.

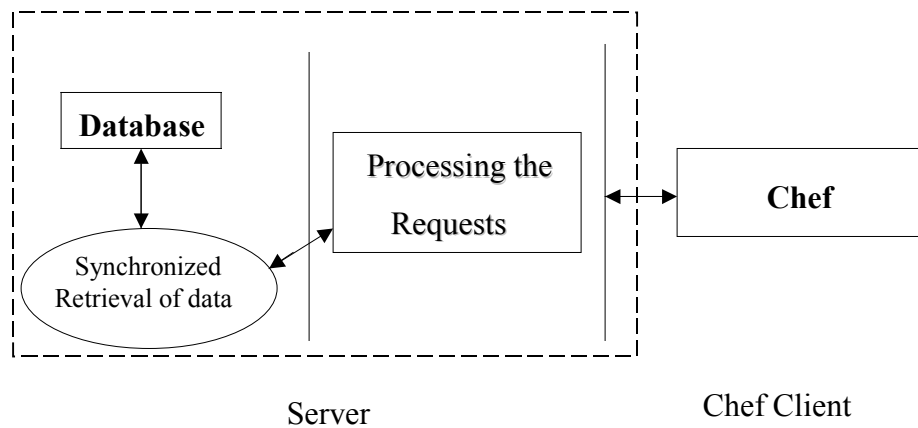


Figure 3.7: Communication model of the chef

Chef is designed to provide the following functions:

1. Display list of items to be cooked.
2. Send confirmation to the server once order is completed.
3. Depending on the size of the queue, send an approximate estimate of waiting time to the server.
4. Provide alerts to the server as an indication to the management for some specific requirements.

GUI (shown in figure 3.8) provided by this module displays the list of items to be cooked by the chef and informs the server once the items are cooked. Whenever it informs server by sending request, it fetches the item to be cooked. Server upon receiving the information from the chef updates the database. When ever a new item comes to its list, it calculates the amount of time it requires to cook that item and sends to the server. Based on the times sent by all the chefs, the server calculates the time required to cook a particular order and informs the customer.

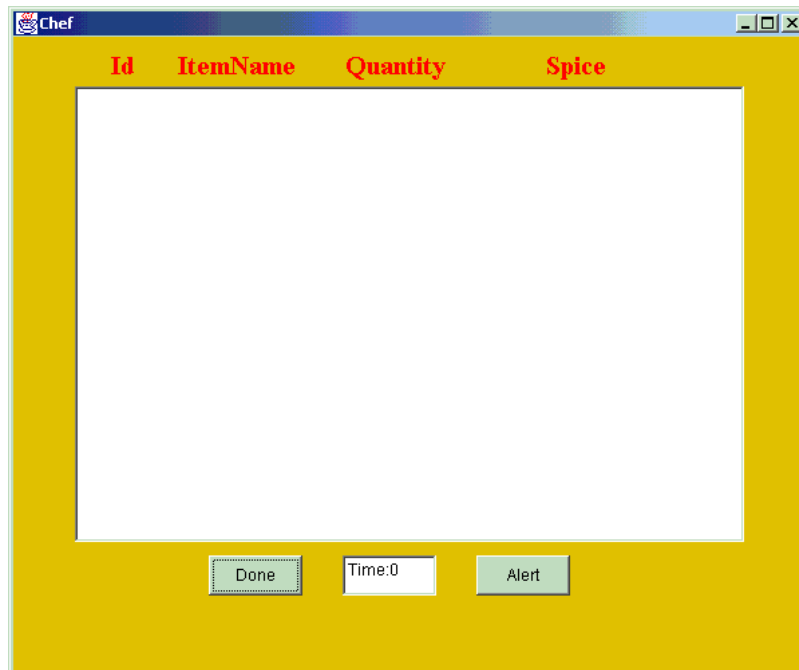


Figure 3.8: Interface provided to the chef to indicate status of the item.

3.2.4 Waiter

This is a graphical user interface provided to the waiters to give an indication of the tables with customers whose orders are completed. A detailed description of its operation is shown in figure 3.9.

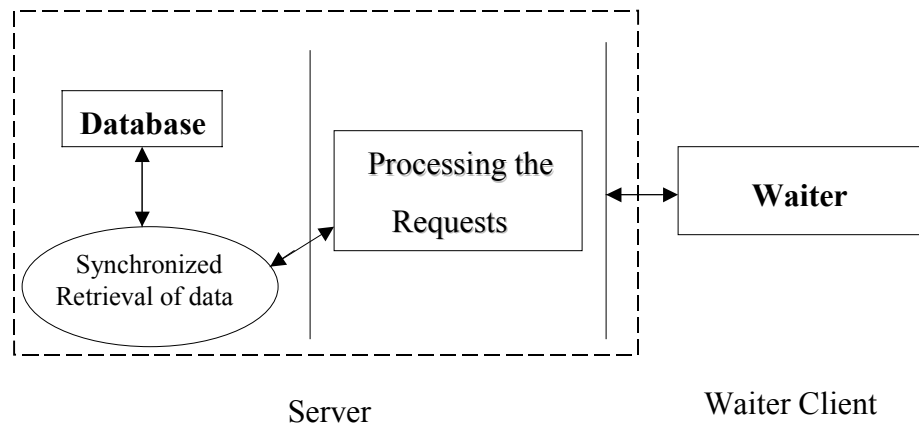


Figure 3.9: Communication model of the waiter

Waiter is designed to provide the following functionalities.

1. Display the floor chart of the restaurant.
2. Calculate the customer orders, which are completed.
3. Update the database with appropriate customer orders and their checkouts.
4. Flash a particular position on the floor chart to indicate orders from that table are cooked.



Figure 3.10: Screen showing the status of the tables whose items are cooked⁶

Waiter is a multi-threaded program interfaced with the image of the restaurant seat arrangement (floor chart). It dynamically updates database with the customer orders and calculates the orders that are completed and blinks that particular position in the floor chart to give an indication to the waiters that the orders from that table are completed.

⁶ The image of the restaurant floor plan is taken from the prohost.

3.2.5 Concierge

This is the table management service provided by the e-Staurant. When a customer arrives at the restaurant, it takes the party size and finds out exact fit for the party. If there are no positions available then it will estimate the amount of waiting time and informs the customer. A detailed list of operations done in this module are indicated in figure 3.11.

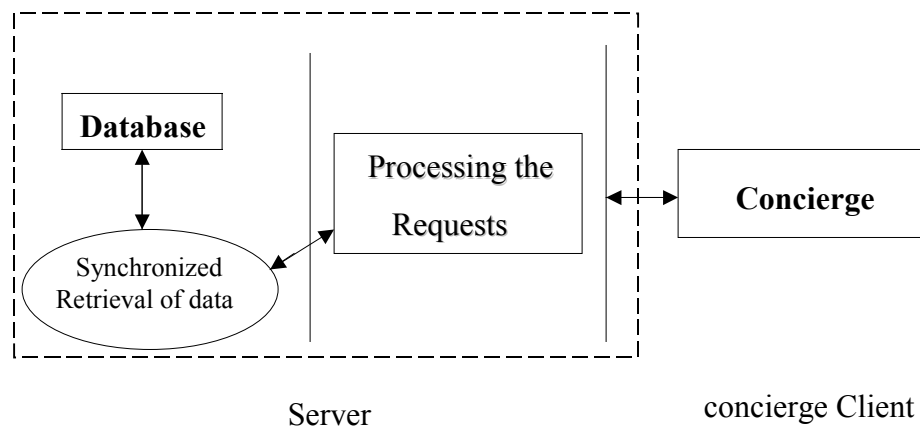


Figure 3.11: Communication model of the concierge

Whenever customer enters the restaurant, concierge sends a request to the server asking for the list of available seats that are fit for the party. Server processes the requests by querying the database through the interface. Concierge will be having the floor plan of the restaurant. After getting the response from the server it- processes the response and finds out the list of tables and their coordinates and changes the color of those positions

in the floor plan. It also provides a user friendly GUI to the concierge to enter the details of the arrived customer and it sends these details to the Server. Server will update these details in the database.

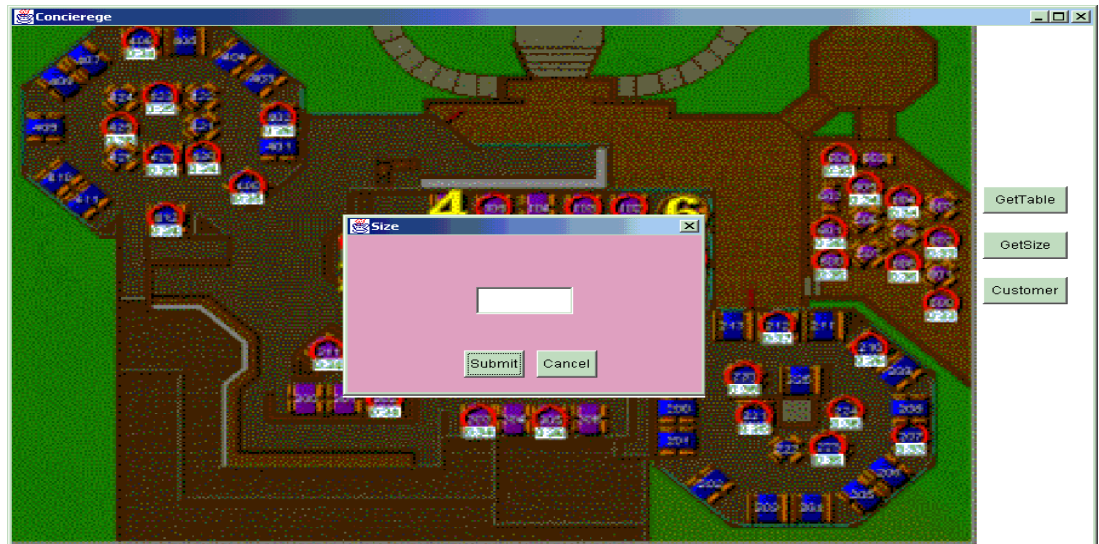


Figure 3.12: Finding out the suitable table for the party arrived



Figure 3.13: Interface provided to the concierge to enter the customer information

3.2.6 Customer

This is a module running on iPAQ device. A detailed description of its operation is shown in the figure. Here end users are the customers whose satisfaction is the prime factor for the success of this e-Staurant software. So here main focus was to provide very good user interface and hide all the communication details from the user and give him an idea that all the data are coming locally. GUI's built in this module are built on handheld device (iPAQ) and care has been taken so that they can be used with windows 32 bit systems.

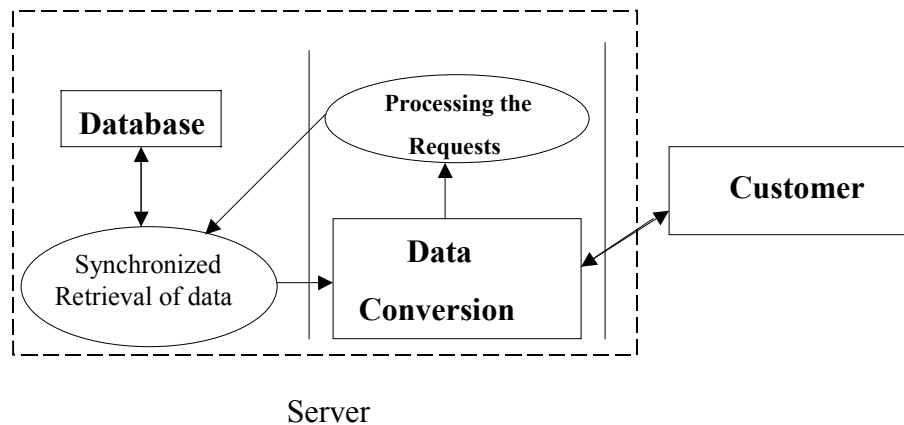


Figure 3.14: Communication model of customer

The major issue to be considered in this module is transfer of data between two dissimilar platforms; we will go into more technical details in this aspect in chapter 4. Customer sends the requests to Server by changing the data format so that the Server can easily

interpret it. The data conversion module in the server will take care of reverse conversion process so that the customer can interpret it properly.

Customer is designed to provide the following functionalities.

1. An interactive menu which includes
 - A list of food items with images and interactive messages describing the food
 - Waiting time of the order
 - Final check when order is completed
 - Providing some features for the entertainment
2. A communication module to get the data dynamically from the server.

A few snapshots of the menu provided to the customer are shown in figures 3.15 through 3.20.



Figure 3.15: Welcome Screen



Figure 3.16: Choosing the type of menu

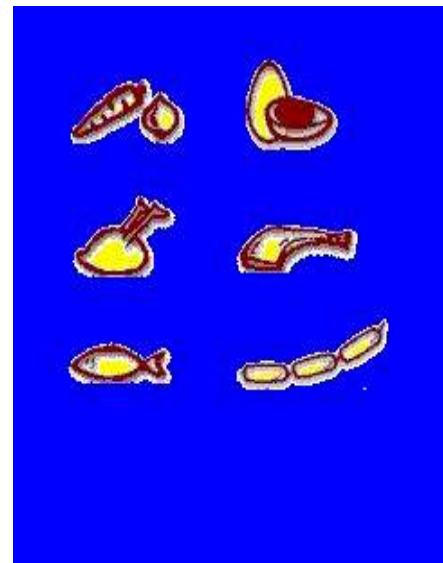


Figure 3.17: Categories of the menu Figure 3.18: Images indicating type of the items

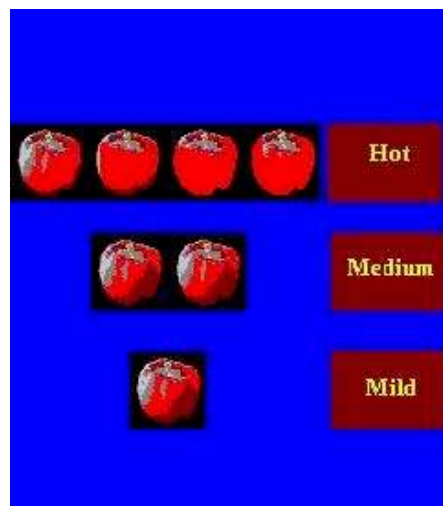


Figure 3.19: Selection of Spice



Figure 3.20: Image indicating waiting time

CHAPTER 4 IMPLEMENTATION: E-STAURANT

This chapter describes in depth the implementation issues of the core components of the e-Staurant. They are RestServer, the server side component, RestaurantMenu, the client process running on the handheld device iPAQ, Webclient, a remote client program, Waiter, a multithreaded program to do table service management. Since the RestaurantMenu is built for iPAQ all its components are Win32 modules.

4.1 RestServer: The Server Side Component

RestServer is the component that handles the communication mechanism of all the clients. As all the clients are targeted to different environments to make the communication simple, Socket communication is used. Entire code of the server is written in java. Server takes requests and sends replies to the clients through Sockets. Writing into the socket and reading from the socket is done by using I/O streams. Most of the requests received by the server are operations on the database. All database operations are handled through an interface called Dbcomponent .It uses a jdbcodbcdriver to connect to the database. DBcomponent makes all the operations on the database synchronized to prevent the problems of stale data and to maintain consistency. DBcomponent retrieves and updates the database by using a set of SQL queries. Database is developed in MS ACCESS. Database is designed to store data related to the customer like his arrival time, party size, credit card details, his food order and his status (checkout/eating/waiting for service/serviced) and menu of the restaurant and allocation of chefs and their names.

4.2 RestaurantMenu: Client Process on iPAQ

RestaurantMenu is the component targeted to iPAQ. This is a win32 module. This is developed in embedded VC++ 3.0 environment using MFC API. Entire module is a dialog-based application. First point of discussion is communication with the server, which is written in java.

A handheld device can be connected to a network in three ways:

- Using a wireless LAN card,
- Using infrared port or
- Using wireless modem depending on the distance range.

Lucent Technologies wave LAN card has been used and a significant amount of work has been done to get appropriate firmware for the card and to configure device to the access point settings and detailed description of these issues is given in Chapter2.

While receiving and sending data from the server it has to take care of byte ordering and character representation, as server is running on NT machine and client on handheld device, which runs on windows CE. When sending data to the server, character conversion is done using a function called **WideCharToMultiByte ()** and while receiving the data it is again processed using **MultiByteToWideChar ()**. Communication is handled using **CAsyncSocket** class.

A CAsyncSocket object represents a Windows Socket — an endpoint of network communication. Class CAsyncSocket encapsulates the Windows Sockets API, providing an object-oriented abstraction for programmers who want to use Windows Sockets in conjunction with MFC.

To use a CAsyncSocket object, its constructor has to be called, and then its Create function should be called to create the underlying socket handle (type SOCKET), except on accepted sockets. For a server socket Listen member function is called and for a client socket Connect member function is called. The remaining CAsyncSocket functions are used to carry out communication between sockets. Upon completion, the CAsyncSocket object is to be destroyed if it was created on the heap; the destructor automatically calls the Close function.

User is also provided with a facility of opening browser from his application. This is accomplished using the function called **ShellExecuteEx(&ShExecInfo)**. During the process of creating some active X controls on the iPAQ, as there is no explicit registry editor all the .dll files, which are to be registered on the device, are done with this function. As there is no proper tokenizing API in MFC a separate class called **CToken** is written to parse the data received from the server into tokens. A sincere attempt was made to make the application utilize entire screen on the device by manipulating the variable **m_bFullScreen** but it is working on the emulator and creating problems on the device. As this component targets the end user who is not much aware of any of the technical issues it is created with lot of attractive GUI's. The challenging aspect of developing GUI for handheld devices is the management of the screen area. Unlike the regular windows programming for desktops there is no concept of window resizing on the device, so whenever a modal dialog is opened by calling DoModal() function of the Cdialog class proper care has to be taken to hide the previous window and the colors on the screen. To make attractive screens and fill them with proper colors and intuitive

images functions like **BitBlt** of device context and **CtlColor** of Cdialog are used. To cover the buttons with attractive images the concept of Subclass in MFC is used.

4.3 Waiter: Table Service Management

Waiter is a multi-threaded program provided for the waitress in the kitchen to give an idea of what orders are ready. The purpose of this component is to give a visual notification to the Waitress that order of a particular table is completed.

All the coordinates of the positions of the tables in restaurant floor plan are calculated and stored in servers database. A unique number identifies each table and its coordinates are stored based on this number. Whenever a customer orders food, server distributes the order among the chefs. Waiter continuously monitors the status of all the items of a particular order and waits for the server to set the status of the items. Once server receives a message from the Chef that the item is cooked it sets the status of the item in the database. When all the items of an order are cooked waiter retrieves the table number from the database. Waiter loads the gif image of the Restaurant floor plan on frame and changes the color of the coordinates identified by the table number.

As the status of the table is to be continuously monitored right from the point customer occupies it, a multi-threaded program is written. An event will be triggered whenever customer occupies the table and this event starts a thread, which forks one more thread to continuously monitor the status of the order. If the customer checks out with out any manual intervention of the waitress the color of the table in restaurant floor plan is changed back to its normal position. The other components of the e-Staurant, concierge and chef are similar to Waiter; they also deal with the restaurant floor plan and the position of the tables in the plan, their code doesn't involve much of threads but the

way they manipulate the restaurant plan is similar to that of Waiter. All the data required by the waiter to find out the status of table and after that status of the order is obtained from server and this communication is done using sockets. Synchronization of operations on database is point of interest in this regard because of multiple threaded access to the database at the same time and this achieved on server side with the interface DBcomponent.

4.4 WebClient

This component is an interface provided to the manager of the restaurant to access and modify the menu of the restaurant and distribute the work among the chefs. It will be using the browser to post a query to the server. This is accomplished with the help of servlets.

Java Servlets are server side components that are analogous in many ways to CGI programs. They handle web requests, returning data or HTML programmatically rather than from a static file. In the context of this project, when a request comes in to a servlet, the servlet invokes the Java interface, DBcomponent present on the server side component and gets necessary information on behalf of the web request. Servlet runs in servlet engine.

Servlet Engine: A servlet engine does the following tasks:

- It loads the .class file in the Java virtual machine running on the server.
- It runs the servlet and provides it with a runtime environment

Javawebserver2.0 has been used as servlet engine in this webclient.

Authentication of manager is done before allowing him to modify the data.

Manager is given a chance to browse through the menu and modify items, and when he is adding items to the menu he will be given option to allot that to a particular chef.

4.5 Summary

All features of e-Staurant have been implemented keeping in mind the features and principles enumerated in Chapter3. The resources of the windowsCE and Java API have been exploited to their maximum extent to provide a very good GUI for non-skilled end users.

CHAPTER 5 EXPERIMENTS

5.1 Introduction

The main motivation of e-Staurant is to perfectly stream line and automate every single operation of the restaurant management to provide best quality of service. The goal of this experimentation is to provide an analytical model for analyzing the impact of e-Staurant. The main factors considered in this experimentation are waiting time (time spent in the restaurant) and throughput (Number of customers serviced in a given amount of time).

5.2 Conditions and Assumptions

To analyze the system we have simulated real time scenarios of the restaurant. To clearly show the impact of e-Staurant, analysis of the traditional restaurant model is also provided.

The following are the set of parameters that are considered when doing the simulation [14].

- The average number of customers in the system (the typical number of customers either waiting in the queue or undergoing the service).
- The average delay per the customer (the typical time the customer spends waiting in the queue plus service time).
- The customer arrival rate (the typical number of customers entering the system per unit time).

- The customer service rate (The typical number of customers the system serves per unit time when it is constantly busy) also called as throughput of the system.

5.2.1 Analytical Model of a Traditional Restaurant

The model indicated in figure 5.1 is based on the following situation. SIRO in the figure indicates serial in random out. When a customer enters the restaurant, he is greeted at point "A" indicated in figure 1 then he is allotted a seat in the restaurant where he waits for the waiter for some time. Waiter comes there takes the order and places it in the kitchen in a queue before the chef. Chef after finishing the cooking places it in

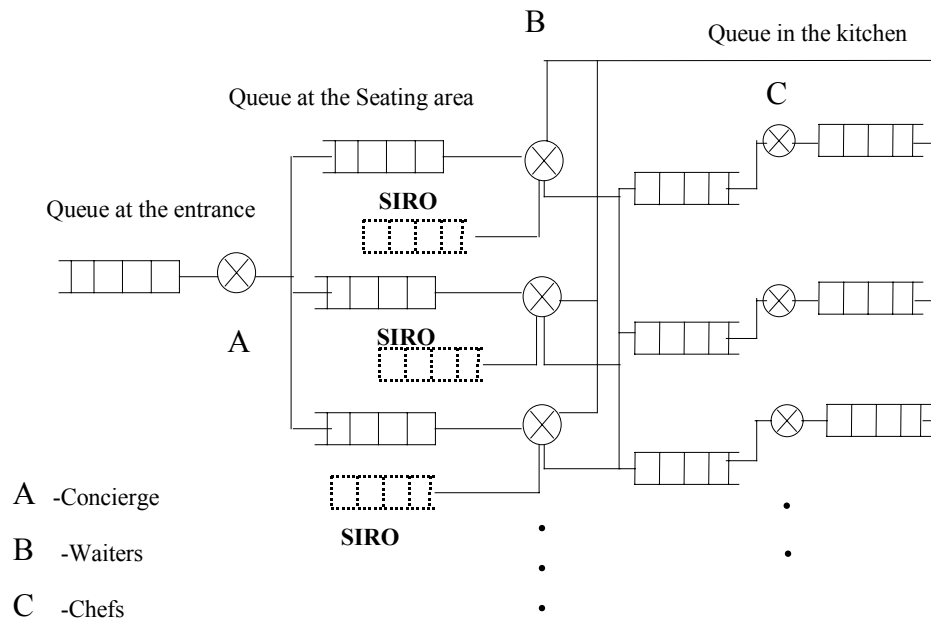


Figure 5.1: Representation of the traditional model

another queue, which is serviced by the waiters. Waiters take orders from that queue and serves them to the customer. Two queues are shown in figure 5.1 at point B (Before the waiters) the queue with solid line indicates the queue of the customers who are waiting for the waiter to place their order, the dotted line queue indicates the queue of customers whose orders are taken by the waiter and who are waiting for the food.

5.2.2 Analytical Model of e-Staurant

The model shown in figure 5.2 is based on the following situation. When a customer enters the restaurant, he will be greeted at the concierge and he will be given a handheld device, which contains the menu. The device indicates whether there are any seats available or if he has to wait for a certain amount of time. Based on the seat availability, he will be entering into the seating area. But his placing of orders is

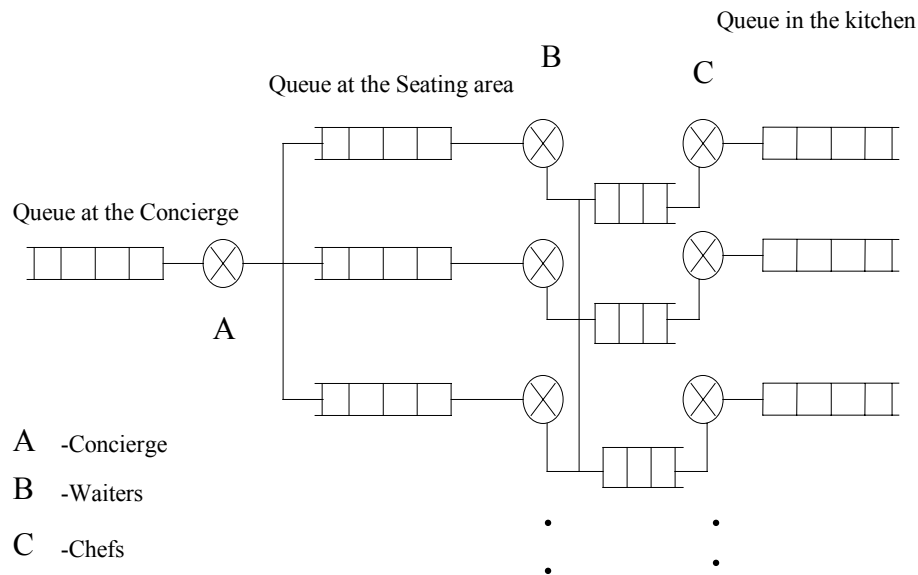


Figure 5.2: Representation of e-Staurant

independent of his arrival into the seating area. Once he gets seated, waiter will come to him directly with the food items.

When compared with the model of a traditional restaurant, the reductions of waiting times in the system will occur at the following points.

- There is only one queue before point B in e-Staurant.
- There is no loop between the waiter and queue before chef.
- Traditional model assumes certain amount of service time (another queue before either chef or concierge) for the completion of service of a customer (getting him the final check). This time is eliminated in the e-Staurant.
- In the traditional model, waiter has to take care of three queues at any point of time whereas in the e-Staurant he will be dealing with one queue at a time, leading to less number of waiters to serve the system. Because of the elimination of the queue of customers waiting for the waiters, restaurant management can take advantage of this time to utilize the waiter for other purposes.

5.3 Simulator

A Java simulator has been developed to simulate the realistic scenarios by making use of the representation given in the previous section. Simulator gives total amount of time a customer spends in the restaurant. A delay analysis plot is given to analyze the amount of waiting time a customer may experience in both the models.

5.3.1 Working of the Simulator

Figure 3 gives a detailed description of the operations performed by the simulator. It has five modules: RestSim, Waiter, Concierge, Chef, and Customer.

RestSim generates customers randomly for certain amount of time (which is the inter-arrival time). The customers generated will be given certain attributes at the time of creation . Those attributes specify characteristics of the customer like party size, type of zone they prefer in the restaurant and the time at which they entered the restaurant. These are utilized by the table management service to seat the customer.

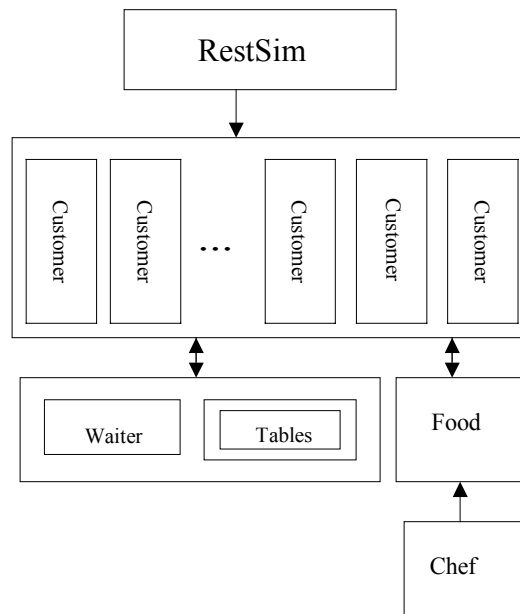


Figure 5.3: Various operations of the simulator

Concierge checks the table management service whenever customer arrives and places the customer accordingly. The function of this module is depends on the type of

the system, if it is old model or traditional model then there is lot of difference in the table management service. In the traditional model customers are served in the order they enter the restaurant. In e-Staurant the attributes of the customer are properly utilized in serving the customer. In e-Staurant if a party of size 4 comes and party of 2 comes, and if table management service says that tables of size 2 are available and table of 4 are not available it will allow party of 2 who came later to take the seat. Tables in the restaurant are implemented as shared objects among all the customer threads generated.

Waiter simulates different conditions of the waiters as represented in the previous section depending on the type of model (either traditional or e-Staurant). For the traditional model waiter does three functions namely, giving menu to the arrived customers and taking order from them, serving the customers who have ordered, getting the food which is already cooked by the chef. Where as in e-Staurant function of waiter is only to get the orders from the kitchen and serve the customers. This is implemented as a shared object on which all the customer threads generated by the RestSim wait.

Chef simulates the condition of cooking the items placed in queue in first in first out (FIFO) order and forming a queue of items which are cooked. Based on the number of items it has to cook it estimates an approximate amount of time and sends it to the server. All the items ordered by the customer are kept in a global data structure called food . This structure is emptied by the Chef at regular intervals indicating that those items are cooked.

Customer simulates the different conditions experienced by the customer in the restaurant like waiting for the table, waiting for the waiter (in the traditional model),

ordering the food, waiting for the order, eating, waiting for the final check, paying the check, and leaving the system.

5.4 Experimental Results

Based on the results of the simulation, the impact of e-Staurant is calculated using the following two factors: delay and load sharing analysis. Delay analysis is a plot between inter-arrival times and waiting times experienced. Load sharing analysis is a plot between number of waiters and load (mean inter-arrival time \div mean waiting time).

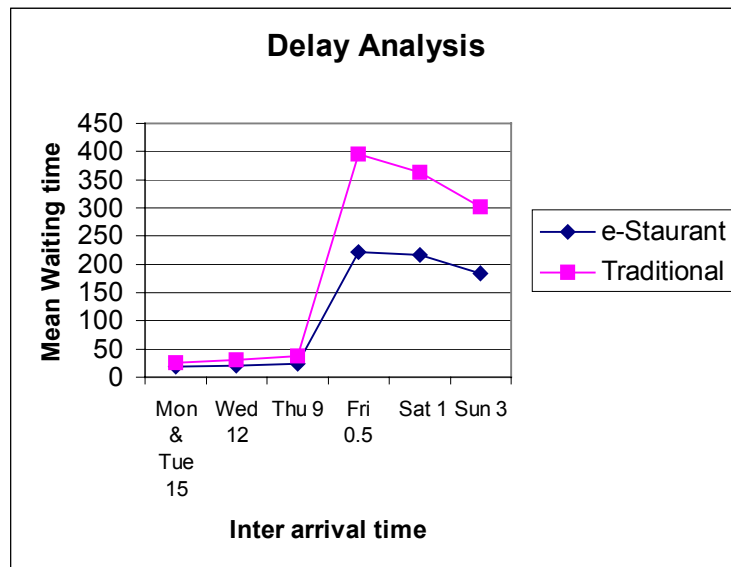


Figure 5.4: Delay Analysis

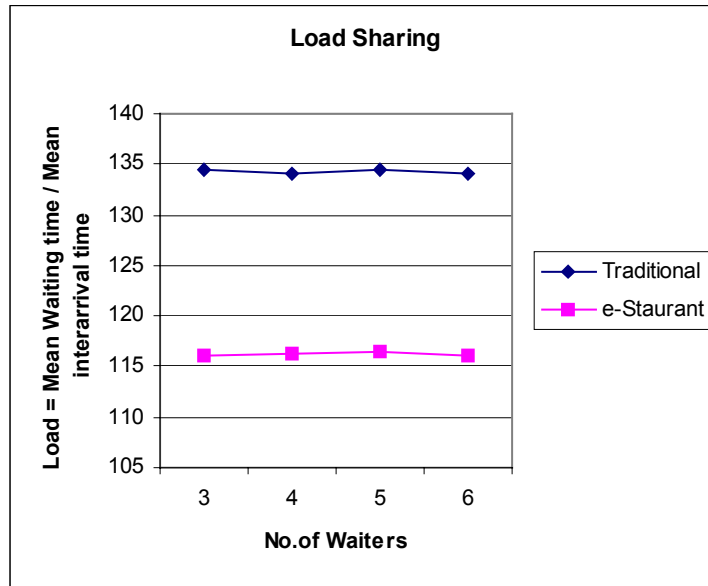


Figure 5.5: Comparison plot for Load sharing.

5.5 Conclusion

Figures 5.4 and 5.5 compare the performance of e-Staurant over the traditional restaurant. Figures 5.4 is the delay analysis of the restaurant and they clearly indicate that at any point of time the waiting time experienced by the customer in e-Staurant is far less than what he experiences in the traditional restaurant. Graph clearly indicates that the impact of e-staurant is more intense when interarrival is very less (For a crowded system on a busy day), which is good sign for the restaurant management. Figures 5.5 indicate the behavior of load shared by waiters on a very busy day of the week (Friday). Load here is Mean Interarrival time/ Mean waiting time It clearly indicates that the load experienced by the waiters in e-Staurant is far less than the load experienced by a traditional restaurant waiter. This is good option for the restaurant management to reduce the number of waiters and efficiently utilize the available waiters.

CHAPTER 6 CONCLUSION

This chapter is a brief overview of the goals accomplished in this thesis. It concludes with a few notions on future work in the area of handheld devices in the field of mobile computing.

6.1 Goals Accomplished

Mobile computing devices together with an ever growing land-based and wireless communication network infrastructure are the existing technical prerequisites for continuous access to networked services. This relieves users from being bound to their desktop computers and lets them spread out into the world. This thesis brings to light the growing importance of handheld devices in mobile computing applications.

e-Staurant is a software infrastructure for restaurant management automation. e-Staurant is an integrated restaurant management software suite that automates table management, chefscheduling, waitress allotment, customer notification, electronic menu and a remote web interface for the manager to modify the menu and chef allotment's. e-Staurant increases control and profitability of restaurant by pleasing customers as well as the management. It pleases the customers by increasing quality of service and reducing the unwanted waiting time and proper utilization of waiting time. It pleases the management by increasing the throughput (number of customers serviced in an given amount of time) with minimum overhead (waitresses, chefs and other staff of the restaurant).

This thesis gives a detailed idea of the factors to be considered for the application development on handheld devices and an overview of the technologies required for making handheld devices wireless enabled. It also gives a brief comparison of the two major operating systems prevalent in the handheld devices world. The third and fourth chapters in this thesis discussed the design and development issues of e-Staurant with an indication of software used in each component. System was designed and implemented keeping in view the bottlenecks of the traditional non-automated models. Some of the technical challenges encountered and their proposed solutions are also enumerated in these chapters.

Finally an analytical model for analyzing the performance and impact of the e-Staurant is provided. Different performance results were shown to analyze the impact of the system over the traditional non-automated models.

6.2 Future Work

The following are the few issues, which we feel one can ponder and make a cause for future work in this area.

Strength of e-Staurant can be further enhanced by applying the concept context-aware computing. Context-aware computing is a mobile computing paradigm in which applications can discover and take advantage of contextual information (such as user location, time of the day, nearby people and devices, and user activity).

e-Staurant can take the advantage of location dependence and can become a good concept for the context aware computing. Instead of restricting the concept of mobile computing to the range of single restaurant, this thesis can be extended to provide location dependent information. When user enters a new location his mobile device can

contact the restaurants in that locality and provide information like type of the restaurants, type of food available and their menu and a provision for ordering the food on the way to the restaurant and find out the waiting time one can experience if he goes to that restaurant.

An option can be given to the user as well as the restaurant to maintain their previous experiences. From restaurant management point of view guest recognition can be made and certain special characteristics can be displayed to please the guests and some special things can be served as a complimentary gifts without affecting the business values. Also, the users can make a quick note of the restaurants and their interests regarding the type of restaurants, which will help them in choosing the best restaurant in a short amount of time.

LIST OF REFERENCES

- [1] Description of Windows CE,
http://msdn.microsoft.com/library/techart/msdn_hndclint.htm, June 2000.
- [2] Operating System Architecture,
http://msdn.microsoft.com/library/wcedoc/apcsdk/sysmgt_1.htm, June 2000.
- [3] Operating System Architecture,
http://msdn.microsoft.com/library/wcedoc/apcsdk/sysmgt_1.htm, July 2000.
- [4] Windows CE Communications Architecture
http://msdn.microsoft.com/library/wcedoc/wcecomm/ndis_4.htm, October 2000.
- [5] A. Bovil, E. Shoemaker, P. Stern, "Portable Chip Architectures,"
<http://www.dartmouth.edu/~pstern/mobile.pdf>, December 2000.
- [6] QuikSpecs of compaq iPAQ Pocket PC H3600 Series,
www.compaq.com/products/quikspecs/10632_na/10632_na.html, January 2001.
- [7] Writing on PDA,
<http://www.zdnetasia.com/products/gspdacentral/story/0,2000009419,20068441-2,00.htm>, January 2001.
- [8] Installing ActiveSync 3.1,
<http://www.cewindows.net/wce/activesync3.1.htm>, February 2001.
- [9] G.H. Forman, J. Zahorjan, "The Challenges of Mobile Computing," Computer Science & Engineering, University of Washington, March 1994.
- [10] A. Joshi, S. Weerawarana, R.A. Weerasinghe, T.T. Drashansky, N. Ramakrishnan, E.N. Houstis, "A Survey of Mobile Computing Technologies and Applications," Department of Computer Sciences, Purdue University, October, 1995.
- [11] P. Bhagwat, I. Korpeoglu, C. Bisdikian, M. Naghshineh, S.K. Tripathi, "BlueSky: A Cordless Networking Solution for Palmtop Computers," MobiCom '99, Seattle, Washington, August, 1999.
- [12] H.W. Gellersen, A. Schmidt, M. Beigl, "Adding Some Smartness to Devices and Everyday Things," WMCSA, Monterey, California, December 2000.

[13] Table Service Management Software, <http://www.jcrsystems.com/prohost.htm>, March 2001.

[14] D. Bertsekas, R. Gallager, Data Networks, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1999.

BIOGRAPHICAL SKETCH

Rajanikanth is a native of Hyderabad, India. Born on November 3, 1977, he attended Osmania University and graduated with his bachelors degree in electronics and communication engineering in 1999. To pursue his master's degree in computer science, he joined University of Florida in August 1999. Upon completing his master's degree, Kanth will join General Electric Medical Systems in Milwaukee, Wisconsin. A sports fanatic, his passions include cricket, table tennis and American football.