

A look into the Palm OS

CS 384 Research Paper
Submitted to: Dr. Christopher Taylor
Milwaukee School of Engineering

By: Kris Berge
2-03-03

I.	Introduction	1
II.	The User Side	1
	A. Performance Expectations	1
	B. Inputting Information	3
	1. The Graffiti Pad	4
	2. The On-Screen Keyboard	4
	C. Viewing Information	4
III.	The Key Features	5
	A. Application Launch Codes	5
	B. The Event Driven System	7
	C. The Memory System	8
	1. Chunks, Databases and Heaps	8
	2. Storage RAM	9
	3. Dynamic RAM	10
	D. The Application Creator ID	11
	E. Conduits and the HotSync Operation	12
	1. What are Conduits	13
	2. Sync Logic	15
IV.	Conclusion	18
V.	Works Cited	19

Introduction:

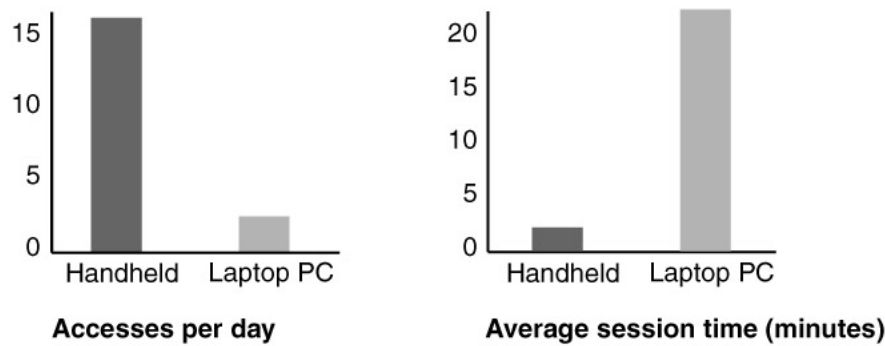
The Palm Operating System was designed for Palm Computing's, now Palm Inc. line of small Personal Digital Assistant (PDA) handheld computers, which premiered in 1992 with the Pilot 1000. Since then the Palm OS has spread to PDAs developed by Sony, Handspring, Symbol and other companies (Foster, forward). Palm Inc. also has continued its line of products including products like the Zire and Tungsten. To date the number of Palm handheld sales has passed the 20 million mark (Palm Handheld Sales Surpass 20 Million Mark). So what has made this operating system so popular with these handhelds? The Palm OS has several key features that make it both fast and easy to use despite the limitations of these small computers. This paper will discuss some of these benefits such as Memory Management and how the Palm OS quickly transfers information and interacts with desktop computers.

The User Side:

Performance Expectations:

The small size of these PDA's requires that their operating system, and all programs related to it, be very small and efficient, because of this the Palm OS has been designed to be compact and work with a small amount of memory and processing power. These small computers are also designed to interface with desktop or laptop computers so that data may be updated and maintained easier. PDA's are designed for quick note taking and easy use, this requires that the programs and operating system respond quickly

and easily to user commands and needs. Figure 1 shows the results of a survey conducted by Palm Inc. on the use of a PDA vs. a Laptop computer. This also shows that on average people use their PDA for short amounts of time compared to a laptop but more frequently also.



Source: Palm, Inc. user surveys.

Fig. 1: Use of a PDA vs. Laptop computer (Palm OS User Interface Guidelines, Chapter 1)

These and more constraints have forced the design of the Palm OS and its applications to be created in such a way to allow a desktop computer to perform many processor intensive tasks and allow the device to preserve its power.

The user also wants a quick response; he or she doesn't want to have to wait thirty seconds for the operating system to boot so that they can make a quick note of an appointment. This means that either the machine must have a very fast boot sequence or have another method of not entirely shutting down. To solve this Palm Inc. designed their OS with three modes of operation: the Sleep, Doze and Running modes.

The sleep mode is what users typically refer to as "off." This isn't entirely true, in fact the device never actually turns off and powers down. This is because the user entered information is stored in power sensitive RAM and not on a physical hard disk or

other permanent media. When in this mode the display, digitizer, the component for sensing screen taps, and other non-essential components are turned off but power is kept on to the main system clock, interrupt generation and the RAM. This means that it must make excellent use of battery power since users expect these devices to go for weeks without charging in some cases.

The Doze mode is designed for when the device is displaying information but not running any instructions. This is the mode that it spends most of its “on” time in. The display, digitizer and other components are active. This allows the device to quickly begin processing information from the user such as a screen tap or from an interrupt triggered event like the system alarm clock.

The Running mode occurs when the device is executing instructions. This is similar to Doze mode in that the PDA typically goes from Doze to Running and back to Doze. The user can send the device into this mode from Doze by providing input, or the interrupts can trigger it automatically from Doze or Sleep mode. Usually PDAs only spend 5 percent of their “on” time in this mode (Foster, Chapter 2).

Inputting Information:

PDAs are designed to function on their own without any additional devices, keeping this in mind there are two primary ways to enter information: the Graffiti pad or the On Screen Keyboard.

The Graffiti Pad

The Graffiti pad works on the concept of text recognition software that the user enters on a small section of the screen. The text the program is designed to read, however, is not the same as what we would write on paper. The graffiti software ignores the idiosyncrasies of an individual's writing style and instead relies on the user to learn a special set of writing that it can easily recognize. While this seems annoying and problematic for the user, the alternative would be a large and complex software recognition program that would take up much more memory and processor power that is not readily available on any PDA. The newest version of Graffiti, Graffiti 2, is designed to use a multi-stroke character set that is more natural and easier to learn for users (Palm OS 5.2 Overview).

The On-Screen Keyboard

The Palm OS also provides an on-screen keyboard to allow the user to enter information. The Digitizer reads screen taps that correspond to specific keys shown on the screen for the character to be entered. While not as fast as a full keyboard, which would be large and difficult to include, this is a quick way for the user to "type" in information or a message (Palm OS Programmers Companion Vol. 1. Chapter 1).

Viewing Information

The concept of the PDA requires that its screen is small and simple; this means that all the features of a desktop computer cannot be included into it. The Palm OS is designed so that the user can customize the information displayed on the screen without

difficulty and get to the most important features easily and quickly. This means the OS and applications will often not display all the features that a program has and simply just display one component. Then when the user needs another feature a screen tap brings it to the foreground.

The Key Features:

The Palm OS differs from other operating systems in many ways. Applications in the Palm OS don't have a main function like in standard c programs, but a PilotMain function is in its place. The User Interface Application Shell (UIAS) is the part of the OS responsible for displaying information on the screen operates inside a single thread. This means that while the operating system is a preemptive kernel the UIAS must wait for an application to finish before gaining control over the display. This also means that unlike typical desktop operating systems the Palm OS cannot display multiple "windows" on one screen. To transfer data between a desktop and the PDA's Palm OS a conduit is created and the data transferred along it. The main advantage of the Conduit is that it isn't dependant on Microsoft's MFC programming and can work on multiple Operating Systems. This section will describe in some detail these features and others.

The Application Launch Codes:

As mentioned earlier applications for the Palm OS must have a Pilot Main function. The purpose of this function is to receive and respond to "Application Launch Codes." Each time the system launches and application it calls this function and passes it a launch code. This launch code instructs the application what actions to perform and

how to do them, whether it be displaying information on the screen or simply performing a calculation in the background (Palm OS Programmers Companion, Chapter 1).

When an application is sent the normal launch code, it gives up control to the event loop. This event loop behaves much like the event driven systems on other platforms and will be covered in more detail later. If the application receives any other launch code, it does not give up control to the event loop but performs a specific action, such as searching its database for a specific entry or opening to a particular entry in its database (Foster, Chapter 2).

The PilotMain function takes in three parameters: UInt16 launchCode, MemPtr cmdPBP and UInt16 launchFlags. The first parameter is the launch code that is passed to the function. Some examples of launch codes are provided in table 1-1 below.

Launch Code	Description
sysAppLaunchCmdAddRecord	Adds a record to the application's database
sysAppLaunchCmdDisplayAlarm	Tells the application to display a specified alarm dialog or perform other lengthy alarm-related actions
sysAppLaunchCmdFind	Finds a text string somewhere in the application's stored data
sysAppLaunchCmdGoto	Goes to a specific record in the application's database
sysAppLaunchCmdNormal	Launches the application normally
sysAppLaunchCmdSystemReset	Allows the application to respond to a system reset

This table demonstrates some commonly used launch codes sent to an application by the system and defines what they mean (Foster, Chapter 4). One advantage of the Palm OS is that there are many defined Launch Codes but if one or several don't apply to a

specific program the author can simply ignore it. The last two parameters are optional and used for loading a specific database and launch condition flags.

The Event Driven System

The event driven system allows the OS to maintain control over the PDA while performing tasks. This preemptive OS allows the user to stop an application while it is running without having to interrupt it. Once the application is started, the control passes to the EventLoop routine. This routine has no parameters but performs a loop that checks for the next event the system event queue. The events are predefined enumerated data types that correspond to what the application should do. When an event occurs it is placed into the event queue and waits for the application to get it. This allows the system to maintain some control over an application while not being overly complex. The four event type handlers which are listed and defined here:

1. **SysHandleEvent** handles system events.
2. **MenuHandleEvent** takes care of menu events.
3. **ApplicationHandleEvent** loads form resources and sets up form-specific event handlers.
4. **FrmDispatchEvent** passes the event to the application's own event handler, or lets the operating system perform default actions for that event.

Each event is handled in the order listed; the System has priority over everything else for acting on an event. Typically events such as Low battery warnings, hardware button interrupts, Graffiti input and the Global find function are handled by this stage. If the system does not use the current event then it passes to the MenuHandleEvent. This would take care of such events like the user tapping a Menu item or tapping the appropriate command to display a menu. The next two managers are the application inside each application itself. Instead of the QT-library's Widgets the Palm OS has forms, which correspond to menus, buttons and other objects. These managers handle

actions such as loading and activating resources and other non-system events (Foster, Chapter 4).

These events allow the application to terminate its event loop when the appropriate event is sent and return control back to the system. This allows the system to regain control from the application without having to interrupt it.

The Memory System:

The Palm OS is designed around a 32-bit architecture. The memory addresses are all 32 bits long and it has basic data types of 8, 16 and 32 bits. This means that it could potentially address up to approximately 4 GB of data addresses. Of course currently no such Palm has that capability but it is there for the future use. The first Palm device actually only used less than 1 MB of memory or 0.025% of this address space (Palm OS Programmers Companion, Chapter 5). Despite that the current Palm devices and their related competitors use typically 16 or 32 MB for memory they have the capability for a large amount of growth.

The Palm OS stores its data into volatile RAM, which requires a small amount of power to maintain data storage. Because this memory is volatile, power usage is critical to a Palm device. This is why it never truly turns off, but only enters a Doze mode. The Palm OS stores everything including itself in this RAM and has divided the memory into two separate logical areas: the dynamic RAM and the storage RAM.

Chunks, Databases and Heaps:

The Palm OS stores all data in Chunks of memory. These chunks are limited in size from 1 byte to slightly under 64 KB in size. These chunks are allocated by the Memory Manager for the applications to use as needed. A database is a collection of chunks of memory used to store data. Each database is analogous to a file on a different operating system. This means each database can be created, altered, deleted, opened and closed just like a normal file. Each database contains the following: name, creator, type, number of records, last synchronization date and the records contained within it. Each record in a database is in fact a Memory Manager Chunk. Each of these records can be interspersed with the records of another database in memory.

The Memory Manager stores databases on a heap structure. A heap is a contiguous area of memory that contains one or more chunks of memory. There is one dynamic heap stored in the Dynamic RAM, and many heaps in the Storage RAM. All the heaps, with the exception of the Dynamic Heap, are non-volatile and therefore aren't affected during a soft reset. The Palm OS assigns heaps a 16-bit heap ID number. This ID number starts at 0 for the first heap and progresses by 1 with each new heap. These numbers are also continuous and unique so that no digit is skipped in the sequence. The values are assigned beginning with 0 on the RAM of the memory card 0 and then the ROM of card 0. This proceeds to the next memory card, if present. The first heap, heap 0, is the Dynamic Memory heap and is volatile. Each time the PDA is reset this heap is reinitialized and all the data inside is lost. Additionally when an application quits, the chunks allocated to it in the Dynamic heap are freed.

The chunks in a heap can be either moveable or non-moveable. This is done so the data can be kept contiguous and defragmented. Each heap manages this by having a

master pointer table. This table is used to track all the moveable chunks of data in the memory of the PDA. When a moveable chunk is allocated, a pointer, called the master chunk pointer, is stored in this table as a handle. When a caller requests the allocation of a moveable chunk, it is this handle that is returned to it and that allows the Memory Manager to maintain a relatively minimized fragmented memory. This means that if the Memory Manager moves a chunk in memory that only the master chunk pointer needs to be updated and the application need not know about it. While current Palm OS devices don't carry multiple cards the support for this feature is available for future use (Palm OS Programming Companion, Chapter 5).

Storage RAM:

This section of RAM is analogous to the data storage memory on a desktop. If the PDA should be reset, this section of memory is unaffected. This is known as a "soft" reset. The Palm OS data manager stores each record in the storage section as a chunk of memory for a database. Each data base contains several of these chunks and the header information for the database. Usually the chunks in a database are related in a logical association such as date book entries or address book entries.

Dynamic RAM:

This is the section of memory analogous to the standard RAM on a desktop computer. During the above mentioned "Soft" reset, the Dynamic section of data is reinitialized while the Storage, code and data, Section remains intact. This occurs in the boot sequence of the device. The entire section of the Dynamic is used to implement a

single heap that provides memory for every application with dynamic allocations. The system uses this Dynamic Heap to provide memory for: global variables, system dynamic allocations (TCP/IP, IrDA etc), stacks for applications and so on (Palm OS Programmer Companion, Chapter 5). The system always reserves this entire amount of RAM used for the Dynamic heap regardless of if it is completely used. The following table shows how the Palm OS 3 to 3.5.

Table 2: Memory Usage for Palm OS (Palm OS Programmers Companion, Chapter 5)

RAM Usage	= OS 3.5 = 4 MB TCP/IP & IrDA	= OS 3.5 = 2 MB TCP/IP & IrDA	OS 3.0 > 3.3 > 1 MB TCP/IP & IrDA (Palm III™)
Total dynamic area	256 KB	128 KB	96 KB
System Globals (screen buffer, UI globals, database references, etc.)	40 KB (OS)	40 KB (OS)	~2.5 KB
TCP/IP stack	32 KB	32 KB	32 KB
System dynamic allocation (IrDA, "Find" window, temporary allocations)	variable	variable	variable amount
Application stack (call stack and local vars)	N/A (see note)	N/A (see note)	4 KB (default)
Remaining dynamic space (dynamic allocations, application global variables, and static variables)	184 KB	56 KB	= 36 KB

Note Starting with Palm OS 3.5, the dynamic heap is sized based on the amount of memory available to the system.

With a PDA that has only 4 MB of memory this table shows that only 256 KB of this memory is allocated for the dynamic section. Of this memory 184 KB of it is free for use by all the applications. The rest is used for the operating system and the TCP/IP stack. The free section of memory is then allocated for storage heaps used to hold nonvolatile user data such as appointments, to do lists, memos, address lists etc.

The Application Creator ID:

The Palm OS identifies individual applications and databases with an application creator ID. This ID is unique to every application and must be registered with Palm Inc. The OS uses the creator ID to determine which databases belong to an individual application. The system info screen uses this ID when it describes the memory used by each individual application. If two applications were to have the same ID, the second installed application would overwrite the first and possibly corrupt all of its databases. These ID's are stored as 4 digit case-sensitive ASCII characters and numbers. One example of such an ID is "Pgam." Palm has reserved all the ID combinations with all lower case letters for its own purposes but has left the remaining possibilities open, with the exception of "pqa ." (PalmSource, Creator ID Database) For this purpose Palm has provided and maintains an open database of creator ID's that can be accessed from their web site (Palm OS Programmers Companion, Chapter 1). This website can be reached at <http://dev.palmos.com/creatorid/>

Conduits and the Hotsync Operation:

Despite the fact that a PDA is a mini-computer all its own, it will from time to time require a desktop computer to link with and manage data. There are several concerns that must be addressed before this communication can occur. Some of the problems that can arise are different operating systems interacting differently, correctly making adjustments to existing records and transferring data without error. These are just a few concerns for the HotSync operation that is performed when a user connects their PDA to their computer. The primary use of a PDA is for portable data entry and viewing, not for high volume data entry or a large amount of complex calculations. This is another use for the HotSync operation that allows the desktop computer to perform many calculations and preserve the PDA's battery. There are four main ways the Palm OS transfer data: direct serial cable connection, an infrared connection, a modem or a network connection.

When a connection is made for transferring data and/or applications between the computer and handheld, the HotSync Manager on the desktop and HotSync Client on the handheld are activated. The Client is an application included with the Palm OS on all handhelds and is the starter of the operation. When the user requests a HotSync on their PDA a signal is sent through the connection to the HotSync Manager to begin the transfer. The manager then in turn calls each of the conduits that are properly installed on the desktop for each application that uses the HotSync. This is followed by the Client allowing access to the required databases that will be updated in the HotSync. Each conduit will then synchronize itself with the handheld individually (Introduction to Conduit Development, Chapter 1).

What are Conduits:

The Palm OS accomplishes this transfer with conduits. Conduits are modules of the HotSync that transfer a specific type of data between the handheld and desktop computer. One requirement for conduits is that they require no user interaction at all. This is so the user can simply start the operation and it quickly will finish, reducing the time for a connection to be broken and allowing for a short amount of time the user must wait. This is also very important because users should be able to perform a HotSync while not at their computer terminal and as such not be able to interact with the desktop. The desktop manager uses Notifier dlls to modify, create and delete corresponding records from the handheld. These dlls server as one end for the conduit, with the databases on the handheld the other.

There are three main types of conduits: Installation conduit, Backup conduit and Synchronization conduit. The first type is used for installing new applications to the device and is one way in communication. The Backup conduit is used for backing up any database entries on the handheld that have their backup bit set. These conduits are intended for use when the Synchronization conduit will not be used for a specific database. The Synchronization conduit is used to adjust, add and delete databases and their counterparts on a desktop computer (Introduction to Conduit Development, Chapter 2). Some of the requirements for conduits in general are: Fast execution, Zero Data Loss, Good Conflict Handling and No User Interaction.

The last criterion has already been discussed, but fast execution is an important feature. This is one of the major benefits of the Palm OS is its fast and easy synchronization. To this end they have set strong goals and requirements for developers

to maintain a fast synchronization. Zero Data Loss applies to a few conditions. This category mainly includes a connection loss, if the cable becomes disconnected the application and conduit must not lose any data. The remaining category, Good Conflict handling, applies to the possibility of if a corresponding record conflicts in its data, such as if it were modified on the desktop and handheld, the conduit for this application or database must quickly and efficiently decide what changes to make. These changes are often specific to the type of information such as address book changes or an appointment adjustment and are decided by the developer. Figure 2 illustrates the way that the information is passed during a HotSync operation and how each applications' data is kept isolated.

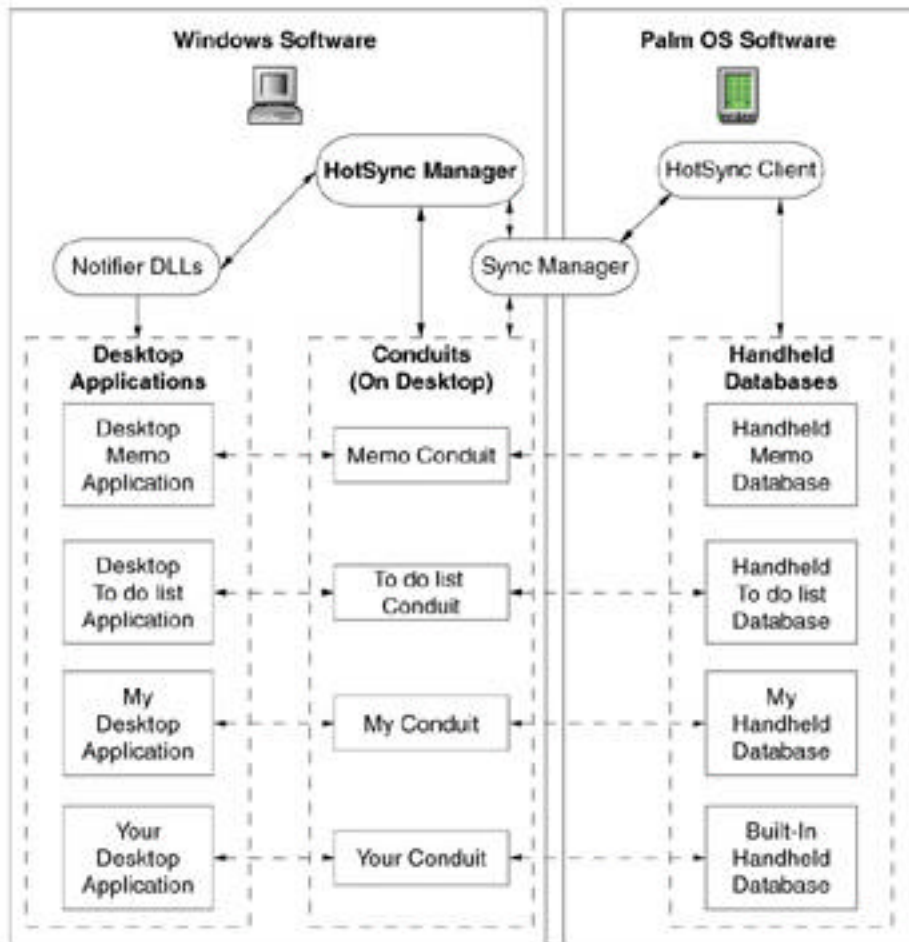


Fig. 2. Illustration of conduit functions and performance during a HotSync

Sync Logic:

When a HotSync is performed the data on both the handheld and desktop are updated simultaneously. This requires that some logic be set up to allow for handling the situations mentioned above. I have included a table from the “Introduction to Conduit Development” book that clearly and easily display the categories and actions taken during these conditions. Table 3 is the included from Chapter 2: Conduit Basics of the above mentioned book.

Table 4: Synchronizing Records, actions and conditions (Introduction to Conduit Basics, Chapter 2)

Handheld Record Status	Desktop Record Status	Action
Add	No record	Add the handheld record to the desktop database.
Archive	Delete	Archive the handheld record and delete the record from both the handheld and desktop databases.
Archive	No change	Archive the handheld record and delete the record from both the handheld and desktop databases.
Archive	No record	Archive the handheld record.
Archive, change	Change	If the changes are identical, archive both the handheld record and the desktop record. If the changes are not identical, do not archive the handheld record; instead, add the desktop record to the handheld database, and add the handheld record to the desktop database.
Archive, no change	Change	Do not archive the handheld record; instead replace it with the desktop record.
Change	Archive, change	If the changes are identical, archive the handheld record and then delete the records from both the handheld and desktop databases.

Handheld Record Status	Desktop Record Status	Action
		If the changes are not identical, do not archive the desktop record; instead, add the desktop record to the handheld database and add the handheld record to the desktop database.
Change	Archive, no change	Do not archive the desktop record; instead, replace the record in the desktop database with the handheld record.
Change	Change	If changes are identical, no action. If changes are not identical, add the handheld record to the desktop database and add the desktop record to the handheld database.
Change	Delete	Do not delete the desktop record; instead, replace the desktop record with the handheld record.
Change	No change	Replace the desktop record with the handheld record.
Delete	Change	Do not delete the handheld record; instead, replace the handheld record with the desktop record.
Delete	No change	Delete the record from the desktop and handheld databases.
No change	Archive	Archive the desktop record, then delete the record from both databases.
No change	Change	Replace the handheld record with the desktop record.
No change	Delete	Delete the record from the desktop and handheld databases.

Table 3 demonstrates that there are many cases which must be taken care of for synchronizing data between the handheld and desktop. If a user were to add an appointment on the handheld and then modify another appointment on their desktop, he or she doesn't want the handheld to ignore the desktop modified one or vice-versa. This is why Palm has created the table for developers to handle these conditions.

The conduits work like pipes in that they allow data to be transferred from one application's database on the handheld to the corresponding applications files on the

desktop. This conduit is isolated from the other conduits occurring at the same time and transfers the data types that it was designed for.

The key features of the HotSync operation allow it to be very efficient and stable for transferring data and make it one of the best attributes for the Palm OS.

Conclusion:

The Palm OS is one of the most stable and well implemented operating systems on the market today. It was and is designed for PDA's to provide quick user access and an easy learning curve for users. The applications are designed to use the single thread for the graphic user interface and minimize power use while not affecting speed and ease of use. The file system is designed to allow the Memory Manager to easily defragment the memory and maintain the databases without losing a record. The memory structure of chunks, databases and heaps allows the Manager to allocate memory quickly and efficiently without a lot of overhead processing or fragmentation. Storing all of the data in RAM also allows for quicker memory access and program execution while efficiently using the battery for extended periods of time. These features have been combined to make the Palm OS very efficient and versatile for use on a variety of PDA's from several manufactures and provide a new field for programmers to develop applications.

Works Cited:

Foster, Lonnon R. Palm OS Programming Bible. Foster City, CA: IDG Books Worldwide, 2000

Palm Handheld Sales Surpass 20 Million Mark. 29 Jan. 2003:

<http://pressroom.palm.com/InvestorRelations/PubNewsStory.aspx?partner=Mzg0TIRFMU1BPT1QJFkEQUALSTO&product=MzgwU1ZJPVAKWQEQUALSTO&storyId=77961>

Palm OS 5.2 Development Overview. 31 Jan. 2003

<http://www.palmos.com/dev/support/docs/palmos50/os52overview.html#grf2>

PalmSource. Creator ID Database. 31 Jan 2003

<http://dev.palmos.com/creatorid/>

PalmSource. Introduction to Conduit Development. Santa Clara, CA: 2001

PalmSource. Palm OS Programmers Companion. Vol. 1 Santa Clara, CA: 2002

PalmSource. Palm OS User Interface Guidelines. Santa Clara, CA: 2002