TASKWEB: ENABLING HYPERTEXT ON THE PDA

by

Janak J Parekh

A thesis submitted in partial fulfillment of
the requirements for the degree of

Bachelor's of Science

Columbia University

October 25, 1999

TR # CUCS-031-99

Columbia University

Abstract

TASKWEB: ENABLING HYPERTEXT ON THE PDA

by Janak J Parekh

Chairperson of the Supervisory Committee: Professor Gail Kaiser
Department of Computer Science

With the recent growth of PDA (personal digital assistant) usage, the amount of information stored in these handhelds has increased tremendously. Much of the data entered on these handhelds bears relationship to each other; they may be related via some common particular element. We call this particular element the *task*, and propose a model that allows for the different pieces of information contained within this task to be *linkable* to each other via the use of N-ary links.

TaskWeb is an implementation of this model for the 3Com Palm series of devices. In its initial implementation, it is capable of link creation, link categorization (into different tasks), link manipulation, and link traversal. It employs a framework that allows it to hook in to the Palm operating system (PalmOS) at a system-call level, thereby making itself omnipresent. Additionally, it stores the links in its own independent database. This enables the product to create links in arbitrary 3rd-party applications and allows it to function completely independent of the applications and their data.

In doing so, TaskWeb bears similarity to a number of systems on the desktop. However, several design issues have been made to accommodate for the limited resources and screen size of PDA's.

Future work in the area of TaskWeb include integration with the desktop, i.e. linking between information on the PDA and information stored on the desktop PC; extending TaskWeb into some intelligent form of caching desktop data and allowing it to be transferred to other desktops; and integration with third-party hypertext systems to extend its functionality and commercial appeal.

TABLE OF CONTENTS

DESIGN ISSUES **5**

SYSTEM ARCHITECTURE **6**

Overall system architecture *6*
Basic functionality *7*
User interface *7*
Database Architecture *8*
Linking interface architecture – the PalmOS "hack" *9*

LINK CREATION **10**

LINK CATEGORIZATION **10**

LINK PRESENTATION **11**

LINK TRAVERSAL **11**

LINK DELETION **12**

CONTRIBUTIONS **13**

FUTURE WORK **13**

Better Generalized Link Handling *13*
Desktop Integration *13*

Chapter 1

INTRODUCTION AND MOTIVATION

Recent growth in the usage of Personal Digital Assistants, or PDA's for short, has been substantial in the last few years. Part of the explanation as to why PDA's have become more and more practical and popular is due to their decreasing size, increased capacity and flexibility, and the ability to transfer the information in the PDA to or from a desktop computer. The most notable of these organizers is the 3Com Palm series of organizers, which are immensely popular due to their small size, ease-of-use and expandability. At this point, users have now have centralized, consolidated access to many different pieces of information that was once spread across pieces of paper.

However, despite the fact that this information is all contained physically within the PDA, different forms of information are stored in different applications or memory areas within the device. Consider the four productivity applications included in the Palm III device:

- Contact/address book
- Appointment/date book
- To-do list
- Notepad

The data within each of these applications stay separate from others. For example, if a new contact *John Doe* is created and an appointment scheduled with

him for next week, I must add him *both* into my contact database and into my appointment book. The two pieces of information stay separate within the organizer; I may be able to trace the connection by knowing about John Doe myself or by using a desktop productivity application, such as Outlook, to maintain the connection, but the PDA does not encode the notion of a link between these two pieces of data. Moreover, many people install 3rd party applications on their PDA's; these applications have no connection to the built-in applications and to any rudimentary relationship features that the built-in applications may have.

While the need for links between two pieces of information is questionable, the value of a link structure becomes useful when many different pieces of data need to be associated with each other; if I have 50 different contacts on a project that need to be related to each other and to group meetings, the utility of being able to link the disparate items becomes greater. Moreover, a given contact may be associated with *different* projects. This prevents tools such as categorization, which on most PDA's restricts contacts to one category. Additionally, PDA applications that have multiple category associations don't support cross-application relationships.

We therefore propose a model where this issue is addressed by the notion of *tasks*, which are collections of *links* that point to *anchors* in data. There is a many-to-many relationship between tasks and links, thereby allowing a link to be associated with multiple tasks and a task to be associated with multiple links (the latter making a task a form of an *N-ary link*). We propose such a linking system instead of a data collection system (for example, an outliner) because the location or type of this information may not be known beforehand, making it difficult to encapsulate. Additionally, the notion of links is very intuitive to the public today due to the World-Wide Web (WWW)'s use of *hyperlinks*.

This proposal bears similarity to hypertext systems available on the desktop, such as **Chimera** [1]. Both of these "open hypermedia systems" allow for linking between arbitrary 3rd party applications, not just a finite set of predetermined applications. The primary difference, however, is that TaskWeb is optimized for the PDA platform, which has different capabilities and restrictions from the desktop, many of which are discussed in the next section. Our system is notable due to the fact that it integrates into the PalmOS to make anchor creation *non-intrusive*, i.e. it does not modify data, yet *omnipresent*, i.e. it is available in every application.

Chapter 2

DESIGN ISSUES AND SYSTEM ARCHITECTURE

Design Issues

The design of a hypermedia system on a PDA device such as the Palm III has a number of implicit design issues.

- *Appropriate user interface.* PDA's have a very small, low-resolution screen, which requires the design of a concise and simple user interface a s per the Palm design manual [2].

- *System omnipresence and independence.* The system must be able to store its data separately, so that anchors are non-intrusive; at the same time, the system needs to be able to present itself in every application, in a minimally multitasking environment (unlike the heavily multitasked environment on PC's), so that users can quickly access its services.

- *Speed.* As per the PalmOS design recommendations [3], people who use PDA's need access to information virtually instantly; they should not be kept waiting for a result, unlike desktop applications which frequently pause while they perform a complex task. In fact, according to Palm Computing, complexity and flexibility should be sacrified to maximize speed and efficiency.

- *Memory usage.* PDA's, due to space restrictions, have less available memory available to them. Additionally, this memory is "static" and "in-place"; unlike desktops and notebooks, there is no distinction between

temporary semiconductor (RAM) memory and permanent storage; only nonvolatile RAM is used. The application must therefore be able to do efficient manipulations within this limited memory space without unnecessarily using up temporary memory.

System Architecture

The PalmOS implementation of TaskWeb (hereafter referred to as "TaskWeb") was designed expressly in mind to take into account all of these issues.

Overall system architecture

TaskWeb has three primary modules:

- **The TaskWeb database**, which is responsible for the storage of tasks, links and anchors;
- **The TaskWeb linking interface**, which allows for the creation of links to anchors and traversal via these links;
- **The TaskWeb maintenance utility**, which allows for creation and maintenance of tasks and maintenance of links.

In the PalmOS environment, a database can only be associated with an application; hence the database is considered "part" of the maintenance utility in the diagram below. We associate the database with the maintenance utility and not the linking interface because the linking interface is not an application but rather an always-available lightweight applet.

Functionality and description of this modular architecture are discussed below.

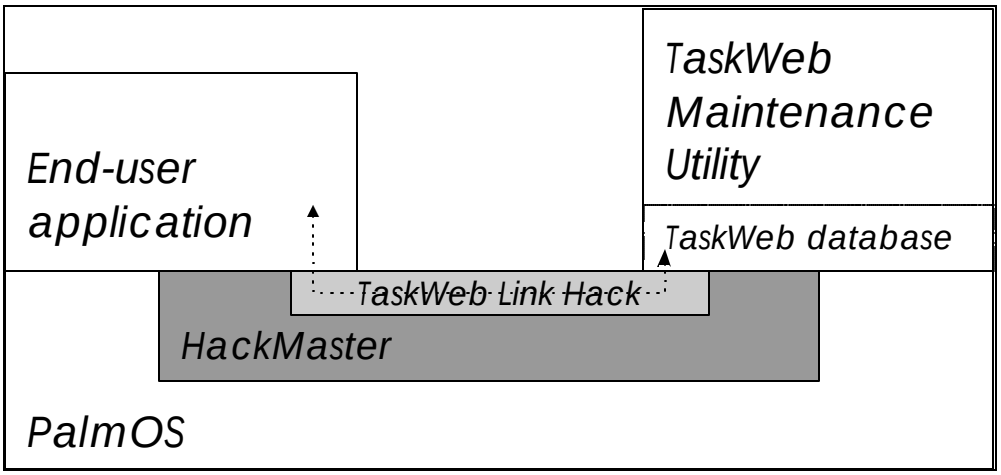

Figure 1. TaskWeb System Architecture.

Basic functionality

The basic functionality of the TaskWeb system is discussed briefly by application.

- The maintenance utility is used to perform housekeeping tasks, such as deleting or creating a large number of tasks. It is not omnipresent.

- The TaskWeb linking interface serves as the standard interface for end-users; the linking interfaces allows for link creation, deletion, and most importantly, traversal. It is omnipresent to allow these tasks to be accomplished while in any application.

User interface

The user interface of both the maintenance utility and the linking interface are designed with simplicity and small screens in mind.

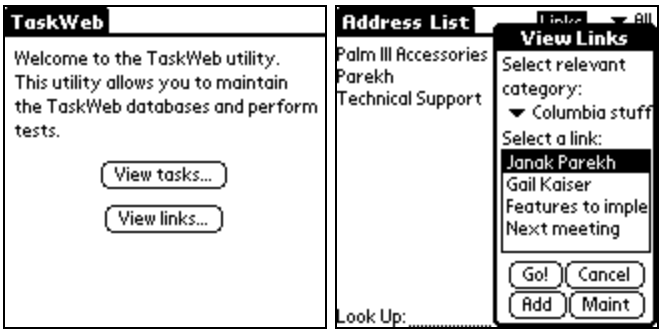

Figure 2. The TaskWeb linking utility and link hack.

In fig. 2, two different screenshots demonstrate the sparsity of buttons within the system, essential as a Palm's device is only 160 pixels by 120 pixels. The linking maintenance utility takes advantage of the entire screen of a Palm device, while the link interface yields functionality while only taking up half of the display.

Database Architecture

The PalmOS provides applications with a structure-based indexed flat-file database system, which allows for the creation and maintenance of databases by applications. To support the many-to-many relationship needed by the proposed linking structure, joins are done at runtime based on index fields. Although impractical for large relational databases, this methodology allows for a flexible, n-ary database architecure.

In particular, two databases are maintained by the system – the task database and the link database. Each database contains references to the records within the other database. When a user requests the links associated with an appropriate task, references to links relevant to that tasks are immediately retrieved, speeding up link resolution. The opposite (links storing containing tasks) is also done to speed up the association of multiple tasks with a given link.

Linking interface architecture – the PalmOS "hack"

As previously mentioned, the linking interface is omnipresent, available within any application. Unlike a multitasking desktop operating system, the Palm OS does not explicitly allow for multiple applications to run at the same time (both PalmOS and Windows CE can employ preemptive multitasking in specialized situations, but neither allow for simultaneous application execution to reduce processor load). Therefore, the linking interface uses the HackMaster API to hook into the low-level system calls and is able to present itself by enhancing such calls to include TaskWeb services. This is the only way, currently, to allow the TaskWeb interface to be present in any application, including both built-in PalmOS applications or in 3rd-party applications.

Since the hack's code intercepts a number of essential system calls to do its work, it needs to be as non-intrusive as possible when it is not being used to eliminate performance overhead. As a result, the linking interface is particularly small and only provides essential services that are needed by the user at that time.

Despite the low-level hooks, hacks are stable additions to the PalmOS, since a utility called HackMaster "controls" the execution of the hacks and ensures only one of them are trapping a given system call at any given time.

Chapter 3

SYSTEM METHODOLOGY

In this chapter, we briefly describe how each of the essential TaskWeb tasks is accomplished.

Link creation

When the user is ready to create an anchor, they have to first highlight the relevant text to be linked to. If this step is not done, the system will try to "guess" the appropriate text by selecting the contextual area around the cursor. In either case, there must be selectable text in order to be able to create this *anchor* (anchors to images are not supported at this time in TaskWeb).

At this point, the TaskWeb linking utility is invoked (by pressing the Links hotspot in the title bar area) and the "add" button is pressed. After the name for the anchor is supplied, the system uses the PalmOS Find API to collect relevant information regarding the highlighted text, such as application name, record number, and other information necessary to jump to this information in the future. This information is finally stored in the TaskWeb link database and control is returned to the user.

Link categorization

At this time, the link is stored into an "Unfiled" task (in keeping with PalmOS category conventions). The user can display the linking interface and choose an

associate operation, which will allow the user to associate this link with one or more tasks.

Link presentation

After links are created, they are presented in several ways:

1. *By underlining the text that was used to create the link.* This appearance is extremely similar to the appearance of hyperlinks on the web. This underlining is done by the linking utility as well – when the user switches to a particular document within the PalmOS, the linking interface determines whether there is linkable content and, if so, displays the appropriate underlines. This method incurs a slight performance hit since the text must be scanned for links before the user can manipulate it, and therefore can be disabled if unnecessary.

2. *By displaying the link in the linking utility.* If the user clicks on the Links button in the title bar area, the linking utility will display the appropriate link in the display. This is particularly useful for tasks such as deleting that link or recategorizing it.

Link traversal

The user can traverse a link in different ways, based on the method of link presentation used.

1. *Clicking on an underlined link:* TaskWeb determines next link associated with the task, switches to the appropriate application, and displays the anchor within that text. In this manner a quick "cycling" of the links within the current task can be performed. Note that if a link is associated with multiple tasks this can no longer be automated; in this case a menu pops

up with the list of associated tasks. After the user picks one of the tasks, the linking interface then proceeds to the next link within the task.

2. *Using the link interface:* This method is particularly useful if the link is embedded in a task that has many associated links – by displaying the link interface window, the user then has a choice of picking one of any of the targets associated with that particular task.

Link deletion

Link deletion is relatively straightforward; the user selects the link to be deleted, and clicks on the Delete button in the maintenance utility. If the user is in the linking interface, a button is provided to jump to the maintenance utility to allow for quick link deletion.

Link deletion affects the link in all categories. If a user wants to "hide" the link from any given category, they should re-categorize the link using the same procedure as the categorization information above.

Chapter 4

CONCLUSION

Contributions

TaskWeb is not intended to be the development of a brand-new technology; rather, it is the result of the effort of porting the framework of hypertext systems to a PDA platform. TaskWeb is unique in how it addresses the design issues of a PDA while maintaining a generalized link structure. However, the future work discussed below demonstrates some of the future potential in this area.

Future Work

Better Generalized Link Handling

Handing generalized links is a difficult task at best. The problem stems from the fact that, to date, the vast majority of Palm applications do not explicitly support a linking structure. There is a discussion under way on the Palm development mailing lists to create a standardized link methodology that developers would agree to support. Such a structure would allow programs that do not have text fields to be linkable; for example, a portion of the Manhattan Subway Map could be an anchor.

Desktop Integration

TaskWeb currently does not handle any desktop integration. The TaskWeb database can be HotSynched onto the desktop computer's hard drive for backup and transfer purposes, but Palm *conduits* can be utilized to provide more extensive interoperability between TaskWeb and the desktop. Conduits are programs written (with help of the PalmOS Conduit Development Kit) and integrated into

the Palm desktop environment to handle the transfer of data between the desktop and Palm in a developer-specified fashion. Such a conduit could transfer data between the Palm device and a TaskWeb desktop applet, or even a third-party linking application unrelated to TaskWeb. The conduit can also be interactive as well; for example, a TaskWeb conduit could handle anticipatory links by verifying them with the user during HotSync, if so desired.

There are several immediately apparent applications that can be developed using the conduits:

- *Linking between Palm data and desktop items.* The PalmOS already allows for primitive "File Link" categories to exist between Address Entries or the Memo Pad and the desktop, but this methodology is not scaleable up to arbitrary PalmOS applications. Since TaskWeb maintains its own link database which is transferred to the desktop computer, the database could be extended to store desktop file links. Such a feature can also give rise to "anticipatory links": a user, when working on their handheld, may realize they need a link to a desktop item; they should be able to do so without being in physical contact with the desktop.

- *Linking between desktop items and desktop items.* This possible extension to TaskWeb would essentially render it as a cache or convenient storage technology. Other forms of data have already been cached on the Palm for convenience (such as Word documents or web pages), so the development of an appropriate conduit and desktop applet would make this a very possible application.

- *Linking between third-party systems and TaskWeb.* A number of linkbase systems aleady exist for the desktop (see Related Work). An interesting research extension of TaskWeb would involve the development of a

conduit to merge these two technologies together to provide a “worldwide” hyperweb that is accessible even when not at the desktop.

RELATED WORK

Anderson, et. al. describe Chimera, a third-party linkbase similar to TaskWeb but for desktop-based applications. Chimera supports a superset of features that TaskWeb supports, including anchors, links (including n-ary links), the notion of hyperwebs, and more. [1]

Tompa, et. al. discuss a data model that allows for an improved utilization of a database architecture; they propose commonality, sets, and a hypergraph model, some of which may be applicable to TaskWeb. [4]

Stotts, et. al. describe a hypertext model and prototype that utilize Petri nets to allow additional flexibility in hypertext systems, especially in the semantics of browsing information. Although TaskWeb has not explored browsing semantics, some of these methodologies may be useful in the Palm hyperweb environment. [5]

Delisle, et. al. have developed a partitioning concept in hypertext systems that allows for a collaborative work environment in hypertext. Such a model could be useful on the desktop side as TaskWeb's tasks could possibly be generalized towards a multiuser environment. [6]

REFERENCES

[1] Anderson, Kenneth M. et. al. *Chimera: Hypertext for Heterogeneous Software Environments.* Proceedings of the 1994 ACM European conference on Hypermedia technology, 1994, pgs. 94-107.

[2] *Designing UI and Program Functionality,* Developing Palm OS 3.0 Applications, pg. 34. © Palm Computing.

[3] *Understanding the Palm OS UI design philosophy,* Palm OS 3.0 Cookbook for Windows, pg. 28. © Palm Computing.

[4] Tompa, Frank. *A data model for flexible hypertext database systems.* ACM Transactions on Information Systems, Vol. 7, Issue 1, pgs. 85-100.

[5] Stotts, David, et. al. *Petri-net-based hypertext: document structure with browsing semantics.* ACM Transactions on Information Systems, Vol. 7, Issue 1, pgs. 3-29.

[6] Delisle, Norman and Schwartz, Mayer. *Contexts—a concept for partitioning hypertext.* ACM Transactions on Information Systems, Vol. 5, Issue 2, pgs. 168-18