

Web-Based Systems for Communication and Scheduling

Hala ElAarag and Robert Hartford
Department of Math and Computer Science
Stetson University
DeLand, FL 32723
helaarag@stetson.edu
rhartfor@stetson.edu

Abstract

Communication and scheduling are very important in today's computer driven world. People are bound to meetings, tasks, and other events that run their daily lives. In this paper we present the design and implementation of a web-based communication and scheduling system. Our goal was to create an efficient usable system that will allow users to stay in touch via e-mail and to provide a means for them to access their schedule from any where that is web accessible whether it is a wired computer or a wireless mobile device. We discuss the problems we encountered in converting our web application that we designed for a wired computer to its equivalent wireless interface.

Keywords: web-based applications, mobile interface, wireless communication

1. Introduction

The high-tech market research firm forecasts that Internet-enabled device shipments will increase from 430 million in 2002 to approximately 760 million in 2006 [1]. People, even if away from their offices, still need to keep in touch with others by email, and create calendars, appointments, reminders, tasks...etc. To accommodate scheduling problems and to prevent having to memorize or write down every appointment, they could use today's web driven technology. Currently, there is a strong drive toward Internet access via mobile terminals. The 2.4 GHz band has fueled the growth of emerging wireless technologies. Wireless personal area networking (WPAN) is led by a short-range wireless technology called Bluetooth. Bluetooth is ideal for applications such as wireless headsets and wireless personal digital assistants (PDAs). Also there are many cellular phones with capabilities for Internet connections. Internet-enabled devices will almost double in 2006. The primary drivers of growth will be mobile handsets, wireless-enabled PDAs, smart phones,

messengers and wireless modems for notebook computers. This allows people to access the Internet almost everywhere. People can have Internet access anytime and anywhere, even in a car or a plane.

In this paper, we present the design and implementation of a web based communication and scheduling system. This system is an integrated online email and calendar system. It would allow a user to check email and create and access their calendars in a user friendly web driven system even if they are on the go.

In this paper, we describe the background requirements for the system. These include the platform, web-server, scripting language, database engine, and standardization for cross browser html/scripting/layout. We discuss the design of the database layout and the database script module. We also present the various aspects of the necessary back-end elements that handle security issues such as password encryption and email access. We present the conversion of our interface to be used on a cellular phone and discuss the problems we encountered in the conversion process.

The rest of the paper is organized as follows. Section 2 presents the implementation of the system for wired computers. In this section we describe the system abilities, the background and foreground requirements, and some implementation issues. Section 3 presents our implementation of the system for a cellular phone. We discuss our conversion method and the problems we faced while converting the system to the wireless device. Finally section 4 presents the conclusion of the paper.

2. System Implementation for wired computers

2.1. System Abilities

Our system had the following goals. It allows a person to create and activate an account on the system, activation methods where chosen so that malicious users couldn't overwhelm the system by creating false

accounts. Accounts are removed after a small period if they are not activated. A valid e-mail account is also used in the activation process to verify users.

Once a user had an account they could create a number of calendars to store their information and appointments. A limitation on the number of calendars a user may create is imposed by the system.

With an account, a user can also setup the system to access a remote mail server to check their e-mail. The system would also allow users to send plaintext e-mail with attachments.

Appointments could have reminders attached to them; a reminder will be sent to the account's email address at the pre-specified interval before the appointment.

The user may change their settings for personalization, e-mail address, their full name, and reset their password if necessary. Users also have the ability to get their account reset if they forget the password.

2.2 Background Requirements

In this section we describe the various parts that would be driving the system, the scripting engine, the database, and the web server. Below are the choices and reasoning for each of the various requirements.

The system should be easily (if not instantaneously) portable to any major operating system. With this in mind a scripting language needed to be chosen that could fulfill the needs of the system and be future upgradeable. Hence, PHP [4] was chosen because it has libraries for Windows, Mac OSX, Linux, and various versions of UNIX. It also allowed the use of almost any web server because if PHP didn't have a native library for the web server it could be used as a CGI application.

The database had to be portable as well and since PHP has very tight integration built in for MySQL [5] and since MySQL runs on all the platforms that PHP does, this was our choice for the database module. Other modules could be written for accessing other databases or even using XML to store data and be easily dropped in place of the MySQL module.

With the scripting engine and the database chosen the last step was to choose the web server. This was a very easy task since PHP integrates very well with many different web servers. Since we had Windows computers at our disposal we installed the Microsoft Internet Information Server module and then setup PHP as a *plug-in for the server*. The web server can be any server that PHP runs under, PHP will handle all the interfacing with the web server and it is transparent to the scripting environment.

2.3 Foreground Requirements

With decisions on the systems driving the back-end solidified the main problem now was how to standardize

the interfaces for the user. The system will be used by people with varying browsers so we looked into what features the various major browsers had in common and what features they had that wouldn't work.

We decided that any client side scripting had to be using JavaScript, as it is the only common language that all browsers support for client side scripting. The client side scripting is important because it allows for the web browser to validate information without having to send it to the server for processing first, this frees up bandwidth and processing cycles.

To ease user interaction all the pages in the system should have a similar look and feel so Cascading Style Sheets (CSS) are being used to handle the formatting and design aspects of the HTML code.

As for the HTML this was its own feat because of the massive differences in Internet Explorer (IE) [6] and Netscape [7] / Mozilla [8]. Both browsers showed strengths and weaknesses and large quantities of incompatibilities. We began narrowing down what types of HTML layout tags we could use and still get the same appearance and effects.

We also originally wanted to use the Portable Network Graphics (PNG) format as the default image format for all of the icons and thumbnails in the system because it provides a high color image format with the ability to have alpha and transparency in the images. Both browsers have support for PNG files but IE can only render high color (24-bit alpha) on a gray background unless a script passes the image through an image filter that only IE 5 and higher supports. This image filtering script was successful at displaying the images correctly but it also seemed to cause a memory leak in IE, it also had the effect in Mozilla of causing the browser to think the page had not been completely downloaded. *With these problems we decided not to use PNG files in favor of the standard GIF and JPEG formats.*

2.4 Implementation issues

In this section we discuss some implementation issues such as PHP interface modules, system layout and database module.

2.4.1 PHP Interface Modules

These modules handle how to accomplish the various tasks the system needs to be able to perform password encryption, sending e-mail, receiving e-mail, and database interface.

Passwords for the accounts on the system could be stored using a one-way hashing algorithm such as MD5; this would allow the password to be authenticated without having to store it in a fashion that could be decrypted if compromised.

However, the password for the e-mail accounts needed to be decrypted to be sent to the mail server, the other option is that the user must provide the password each time they wish to check e-mail. After careful research and trials we determined that encryption of the e-mail account password could not be accomplished by any built-in PHP library, although it does have an encryption library available it is only included in the versions for UNIX, Linux, and Mac OSX. There is a version that works with an older version of PHP on Windows but it no longer works with the current version.

In our research process we found that the company that now maintains the PHP engine, ZEND, has a set of classes for PHP in the PHP Extension and Application Repository (PEAR) library [9], one of which is an encryption library that makes use of the Tiny Encryption Algorithm (TEA). The library uses 128 bit keys and is based on a version written by David Wheeler and Roger Needham of the Cambridge Computer Laboratory. It was also placed in the public domain [10] and is licensed under the PHP 2.0 license [11]. TEA is a Feistel cipher with XOR and addition as the non-linear mixing functions.

E-mail retrieval is handled by a precompiled PHP module called IMAP, which stands for Internet Message Access Protocol; this module was chosen because it has built-in support for communication with: Post Office Protocol Version 3 (POP3), IMAP, and Secure Sockets Layer (SSL) communication with both. The IMAP library provides a standardized interface to the above mentioned mail servers.

Sending mail was more difficult as messages with attachments either need to be sent directly to the mail server using raw sockets or extending the use of the built-in mail function of PHP. The mail function allows for composing simple text messages and can handle multi-part MIME, Multipurpose Internet Mail Extensions, formatted messages but the message must be manually encoded and have the headers pre-constructed. We followed the standards in the RFC (Network Working Group Request for Comments) papers for protocols for receiving and sending mail, IMAP RFC2060 [12] and POP3 RFC1939 [13] for protocols for receiving mail, and MIME 1521 formatting [14] and Simple Mail Transfer Protocol (SMTP 2821) [15] for sending mail.

The database module was very easy because the MySQL module is compiled directly into the standard builds of PHP and is easily added if not pre-compiled.

2.4.2 System layout:

In this section, we describe the design and layout of the system. To enhance user usability of the system, we chose to have a layout similar to Microsoft Outlook with a task bar on the left of the screen a title area to show

where you were in the system and the main body window to handle most if not all of the user interaction and display.

Upon logging into the system a user is presented with the "Look at Today" screen which will dynamically display upcoming events on the calendar that was chosen as the primary calendar, and the current message counts in the e-mail accounts the user has allowed the system to store a password for. It will also display a list of tasks that the user has created. Events and tasks will change color and formatting to show extra information about them, such as: importance, past-due, and completion state.

The user can select tasks from the taskbar to: View Calendars by the day, week or month, View Tasks, View Reminders, Check E-mail, Configure Options, and return to the "Look at Today" screen. The user can also edit the calendar and print it using a printer friendly format.

The control panel allows users to change settings towards their calendars, e-mail settings, appearance of the system, and other various settings. Administrative users can also use the control panel to change system wide settings, manage user accounts, and change user privileges.

2.4.3 Database module

An important implementation issue was how dates and times are stored in the database. Originally, we had dates and times stored as separate string data in the database specification. To speed up database and rendering calls, we redefined the database module in the appointment storage and retrieval. The tables containing the appointment data were redefined to handle the data as UNIX time stamps. The links binding an appointment to a specific calendar were also moved from a separate table into the actual appointment table as too many table joins were being performed to find appointments on specific calendars by the calendar ID.

3 System Implementation for wireless devices

In this section we present the conversion of our system to be used on a cellular phone with Internet capabilities. We discuss the implementation of the wireless interface and the problems we faced during implementation.

The Wireless Application Protocol (WAP) was started by the wireless phone manufacturer Ericsson in 1997 as an attempt to further adoption for using mobile devices to access information, applications, and services. Ericsson created a consortium called the WAP Forum [16] that originally consisted of Ericsson, Motorola, Nokia, and Unwired Planet. The Wireless Markup Language (WML) retains several of the features of standard HTML formatting, such as bold, italics,

alignment, tables, images, and forms. One of the features added to WML is the concept of a deck and cards as opposed to the pages used by HTML, a deck is similar to a normal page and cards represent various sections in a document.

WML also has one other difference with HTML, it is an extension of Extensible Markup Language (XML) with a root element of <wml> and the deck and cards being sub elements of the WML root. In the latest version of HTML there has been a move to adding XML interoperability by redefining the standard HTML tags to be XML compatible. This involves setting the case of the tags, formatting for tag parameters, how tags open and close, and how tags should be nested. With the majority of HTML documents not being in a form that is XML compatible, the task of converting documents for use in WAP systems raises many problems.

Designing web applications for mobile devices has several problems that must be addressed:

- Screen Size / Layout
- Input
- Memory Size
- Wireless Network Bandwidth / Frequencies

In addition to the inherent problem of not being able to test the content on all devices, and not all devices offer a software development kit to test their site with [19]. This is where micro-browser emulators and micro-browser SDKs come in. These emulators and SDKs allow a developer to access their site and to debug problems with their WML markup. The problem with the choice of SDK's or micro-browsers is that like the actual devices themselves many of these software programs do not allow for direct access to the Internet and do not support all of the same functionality, such as scripting and images.

In the conversion process of our Web Calendar System, we chose to use Opera [20] and the Pico Internet Micro-browser [21]. This is because they are both first and second generation WAP compatible browsers and do not require having a wireless gateway setup to interpret the requests. An example of this is the Nokia Mobile Toolkit. Opera was primarily used to view debug messages from PHP when there were runtime errors during script processing as the errors did not conform to XHTML standards and would cause Pico to not render the document.

Metter and Colomb broke down the process of conversion into four methods [17]. However, these methods do not apply well to converting web applications where the data is dynamically generated on the fly by the server like the system presented in this paper. There were many problems implementing the Web Calendar System's mobile interface.

If the code behind the page is sufficiently separated from the display of the page, then the scripting of the

application can be maintained in its normal format but only the code for display is modified. This is similar to the View/Model/Controller methodology, the View is the page that displays the data from the Controller and the Model, the Model is the database engine, and the Controller is the scripting behind the page.

With this design it was relatively easy to convert the site after initially designing the WAP interface using WML/HTML and testing it with Opera and Pico. Although some features had to be removed in order to facilitate the mobile implementation.

Logon password encryption had to be removed as the clients did not support WMLScript, and because the encryption for the HTML site was handled using JavaScript. This functionality could be later implemented using a more advanced SDK and micro-browser emulator to allow the conversion of the MD5 hashing code from JavaScript into WMLScript. Also in general with most new mobile devices and communication protocols there is built-in channel encryption done by the security network layer in the WAP protocol stack, this enables encrypted communication between the device and its base station. While this lowers the risk of the communication being intercepted between the mobile device and its base station, the communication between the base station and the Internet host could still be intercepted unless a secure communications layer is being used, such as SSL.

Due to the differences in designs on mobile devices, images and icons should not be used as a primary means of navigation on a mobile enabled site. Some mobile devices have no ability to display images and if they were used as the primary means of navigation on a site these devices would not be able to navigate. Therefore images and interface icons were removed to limit the amount of data needed to be transferred to navigate the site and to provide increased compatibility across a wide range of devices. The iconic interface was changed to a bulleted listing that can be scrolled through using a mobile devices navigation keys. This can be seen in Figure 2 and can be compared to the framed interface that uses images and icons in the HTML interface in Figure 1.

The control panel functionality was removed on the mobile interface due to the amount of data and forms in this section. The user modifiable settings primarily affect the display and functionality of the HTML interface and thus have direct affect on the functionality of the Wireless interface. The user administration system was removed from the mobile interface due to the sheer quantity of data that would be transferred for a largely populated system; this functionality would need to have been broken up so that the data transmitted to the mobile device did not exceed the abilities of the device or its network. With the interface broken into smaller sections the administrator would have to perform too

many inputs to perform any actions on the user list. The formatting would also be problematic with the amount of information needed to be displayed to manage the user accounts and the system tasks.

E-mail attachments were also removed due to the nature of the environment, mobile devices primarily don't have any secondary storage systems to allow them to send or receive attachments, let alone process data files that were formatted for use on PCs.

The following features were modified from their original format to better fit the mobile environment.

- The majority of the data that was originally displayed in a tabular format was redone in a bulleted list for better display

- Incoming e-mail messages are parsed for HTML tags and they are removed to prevent client errors from poorly formatted HTML in the message
- The navigation system was modified into a hierarchical tree like structure that extends from the main menu into the calendar and e-mail menus and then into their sub functions. This system is navigated like folders in a directory structure

Below are some images of our system for wired computers and the corresponding interface for a wireless device to demonstrate the conversion process.

Login Screen

WEB CALENDAR SYSTEM

Welcome, Please login:

Username:

Password:

Login without java encryption
(Please note this means your password WILL be transmitted over the network in plain text!)

Other options:

- Switch to SSL for secure connection
- Create a New Account
- Send Email to Site Administrator
- Information on installing SSL Certificate
- Download the PIXO micro browser

Figure 1: HTML Login Screen



Figure 2: WML Login Screen

Main Menu

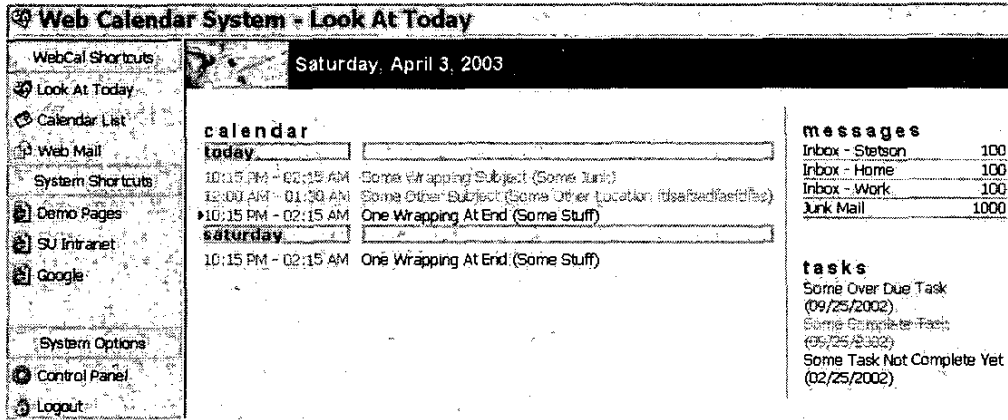


Figure 3: HTML Main Menu

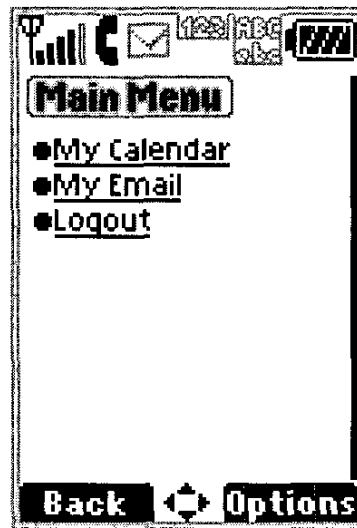


Figure 4: WML Main Menu

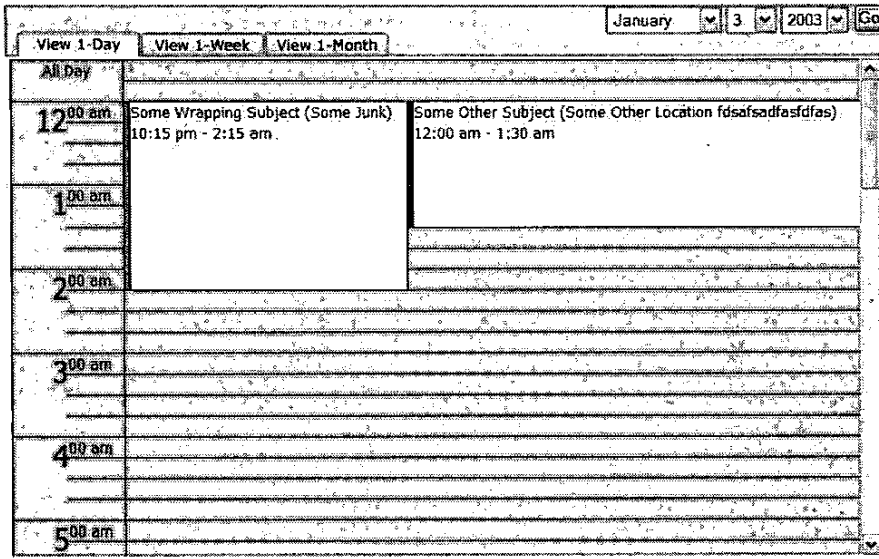


Figure 5: HTML Calendar View events- Day view

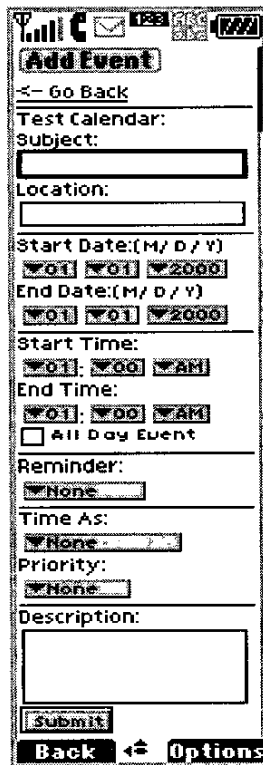


Figure 6: WML Calendar Add event

4. Conclusion

In this paper, we presented the design and implementation of a web-based communication and scheduling system for wired computers. We also implemented an interface to allow the system to be accessed by a mobile device. Converting, maintaining, and interfacing mobile devices into the Internet and web applications is an important priority to allow the data and applications stored on the Internet to be available by any means, anytime and anywhere. We discussed the problems we faced in converting the interface for a mobile device. The major problem is that the web has been standardized for wired computers not wireless communication devices; many of the standards of the Internet don't have a place in mobile communications. We present our novel approach in converting dynamically generated web applications to a wireless interface.

References

- [1] Cahners In-Stat, "Demand Increasing for Mobile Internet Access Devices-Handsets Represent Primary Growth Driver", June 2002.
- [2] Microsoft Outlook
<http://www.microsoft.com/office/outlook/default.asp>
- [3] Apple iCal – <http://www.apple.com/ical/>
- [4] PHP – <http://www.php.net/>
- [5] MySQL – <http://www.mysql.com/>
- [6] Internet Explorer -
<http://www.microsoft.com/windows/ie/default.asp>
- [7] Netscape – <http://www.netscape.com/>
- [8] Mozilla – <http://www.mozilla.org/>
- [9] PEAR – <http://pear.php.net/>
- [10] Public Domain -
<http://www.infoplease.com/ce6/society/A0840435.html>
- [11] PHP License - <http://www.php.net/license/>
- [12] IMAP RFC – <http://www.ietf.org/rfc/rfc2060.txt>
- [13] POP3 RFC – <http://www.ietf.org/rfc/rfc1939.txt>
- [14] MIME RFC – <http://www.ietf.org/rfc/rfc1521.txt>
- [15] SMTP RFC – <http://www.ietf.org/rfc/rfc2821.txt>
- [16] WAP Forum – <http://www.wapforum.org/>
- [17] Metter, Marcin and Colomb, Robert; "WAP enabling existing HTML applications", Proc. AUIC, pp. 49-57, Feb 2000
- [18] Agrawal, Prathima and Famolari, David; "Mobile Computing in Next Generation Wireless Networks", Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications, pp. 32 – 39, 1999
- [19] Niskanen, Pekka; *Inside WAP*, Addison Wesley Publishing under Pearson Education Ltd., 2001
- [20] Opera – <http://www.opera.com>
- [21] Pixa Internet Microbrowser –
http://developer.pixo.com/downloads/download_thanks.html