Proceedings of the APAN – Network Research Workshop 2012

REST-based Offline e-Mail System

Gihan Dias, Mithila Karunarathna, Madhuka Udantha, Ishara Gunathilake, Shalika Pathirathna and Tharidu Rathnayake

University of Moratuwa, Sri Lanka

E-Mails: gihan@uom.lk; xxmithilaxx@gmail.com; madhukaudantha@gmail.com; jaimgunathilake@gmail.com; pspathirathna75@gmail.com; tharindu.ratl@gmail.com

---

Abstract: Over the years the Internet has grown from a research tool to a worldwide communication medium. One of the applications that has grown up with the Internet is e-mail. e-Mail has become an indispensable tool for both corporations and individuals, and web-based e-mail systems have become very popular. However, a major problem with web-based email is we cannot access them when not connected to the Internet. We have built an off-line web-based e-mail system to overcome this issue, and to provide fast response even over slow connections. This system is based on Representational State Transfer (REST) and maintains HTML5 local storage to store mail and meta-data in the browser without installing any plug-ins. The system records all user actions locally and synchronizes with the server when connected to the Internet.

Keywords: REST; offline; e-mail; webmail; HTML5; IMAP.

---

1. Introduction

Email, which is one of the most popular communication mechanisms of modern world, can be divided in to two different categories based on the client side, namely, webmail systems and local applications. Webmail systems, in which the emails are accessed through a standard [12] web browser, are popular because users can access their mail from any computer. However, the main disadvantage with webmail is that it requires a network connection to the e-mail server in order to operate. Thus, when a network connection is not available (e.g., when traveling or outside your office), the email service is unusable.

Even if an Internet connection is available, it may be slow, or high-cost, e.g., over a cellular network, or when roaming.

As e-mail is not a real time system, a user should be able to use it even when the network connection is not available, and many local e-mail applications provide this feature. Our REST-based Offline E-mail System is an open source, offline enabled webmail system, which provides capabilities to use e-mail services regardless of the availability and quality of a network connection.

Traditionally, e-mail systems store messages in files and provide an IMAP interface. A webmail system typically consists of a separate server which accesses the e-mail server over IMAP, and converts messages to html to be displayed on a browser. By providing messages over a REST interface, which can be accessed directly by a browser, the necessity for a separate webmail server has been eliminated [12].

In our solution we have introduced offline web based capabilities with using HTML5 technology [9]. We use HTML5 local storage [10] to store data and events in offline mode. This also allows faster access to messages, as they can be retrieved from the local cache.

All emails are stored in the server database [11]. By using a database rather than a file based system, we improve the performance by overcoming the limitations such as data separation and isolation, data dependence, data duplication, lack of flexibility in organizing and querying the data.

2. Approach

2.1 Current Systems

Several systems have been built to provide improvements over traditional e-mail systems. Several companies offer email server products based on database storage [11]. DBMail offers programs that enable the possibility of storing and retrieving mail messages from a database [1]. It claims that the system performs faster queries, scales better, and provides better flexibility than file-based storage, but it offers little reasoning or evidence to support these claims.

Openwave Email Mx (formerly Intermail) is a product from Software.com which uses Berkeley DB, an embedded data store, to archive messages [1]. Courier and Cyrus are common IMAP servers which are used today. Both of them use maildir (file based structure) as the mail storage. Dovecot is another IMAP server which uses both mbox and maildir for storing email. The major disadvantage of this system is file locking: the mail server must be careful to prevent changes to the mailbox by the user being made simultaneously with changes made by the server. Otherwise, the mailbox file may be corrupted, which may result in the loss of the messages. The Courier mail server's IMAP, POP3, and webmail servers talk to maildirs only.

Web services interface is new in mail services, so existing sources are not much available and standard protocols are not defined. However, some well-known e-mail systems use REST

interfaces for their systems. Among these Zimbra REST API has developed for email and calendar server plus much more; it just like a next-generation Microsoft Exchange server [4]. Zimbra exposes its data via a REST API. This is an approach for building application services that make resources available via a URL. They provide webservice interfaces to series of their services such as email, contacts, calendaring, sharing and document management plus mobility and desktop synchronization.

Zoho Mail also supports offline mode. This means we can access our email when not connected to the internet; this functionality is built using Google Gears. To use Zoho Mail in offline mode, we need Google Gears installed on our browser. In Offline Gmail (Google Gears) we don't need to install additional software other than Google Gears and you can continue to use Gmail's familiar interface but still we are not running in web browser and still he have to install Google Gears. Google Gears project was terminated in 2008 with the introduction of the offline storage feature in HTML5.

2.2. Offline capability

We can find offline cable email clients application and some web user looking offline featuring email system such Google gears. But our Offline Email System is able to run in normally web browser without installing any plugging to browser or computer. We make this feature to our system by using HTML5 [9], newest technology. To run application or web page in offline or without internet connection we have to look main two parts how to make user interface to run in offline and also how to handle data and event in user in offline. There are two resources of data, one is client side and server side so whenever user goes online my system able to synchronize with any data misplace or damage. For that we have been modeled restful web service for to communication on information HTML5 local storage [10].

2.3. Server Side of the email system

The server side of the system consists of three modules:

1. Database for storing emails[13]
2. Acceptance of incoming messages and
3. REST Web Service.

The database [11] is designed to avoid redundant storage of content. Each MIME part of an e-mail store in separate table with a unique hash value which is generated using SHA-1. So if there is same content in different emails, we need to store that content only once.

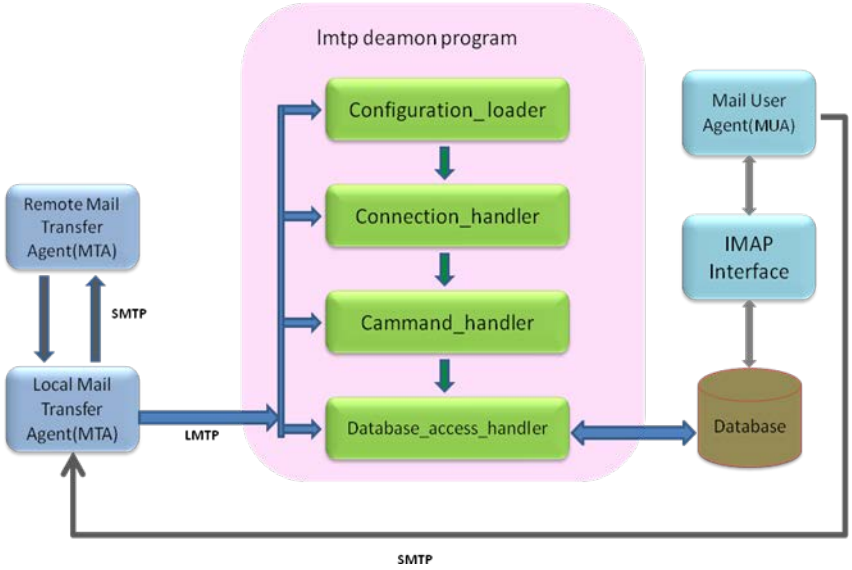

Figure 1. Database integration for IMAP.

The lmtp deamon program accept lmtp command. It split email and insert to the database.

2.4. REST web service

Our system has a REST-ful web service [7] interface running on HTTP protocol and user can consume service without using other protocols such as IMAP. REST service [6] access points are defined analogous to all relevant IMAP commands.

We have also implemented a specific set of functions for offline mail client sync purposes and others are common to use any client program [13]. REST based mail server interface give API for mail access for mail clients as well as to mobile/PDAs/iPods via HTTP.

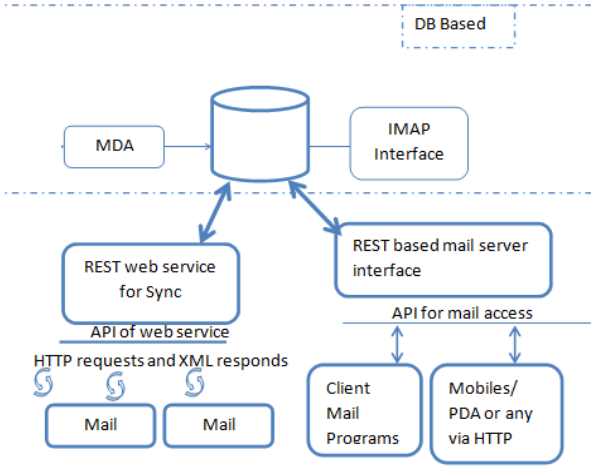

Figure 2. Integration of modules in server side.

In REST we can call particular email and its particular part even so it improves our speed and make less network traffic. REpresentational State Transfer (REST) is software architecture for distributed systems such as the World Wide Web. In here we have used REST for emails. It helps scalability of component interactions, generality of interfaces and independent deployment of components. Here are some basic REST API[7] in emails.

Table 1. Basic email REST API Resources.

Resource	*Description*
GET emails/{username}/inbox	Returns the most recent 25 e-mail headers in particular users's inbox
GET emails/{username}/inbox/{email_ID}	Returns the e-mail body with the list of MIMEs if it is exist
POST emails/{username}/Draft/{email_ID}	Email will saved to Draft and request body will contains the email header and body
PUT emails/{username}/Draft/{email_ID}	email draft will be updates.
DELETE emails/{username}/Draft/{email_ID}	email will be moved to trash folder

REST API can be called from any rest supporting client application. In particular it supports HTML5-enabled mobile browsers such as in iPads, tablets etc.

2.5. Client side with HTML5

We have provided a plug-in-free offline webmail for first time. The client software, written in HTML5, comprises three modules:

1. HTML5 Local Storage [10],
2. Offline HTML5 Web User Interface and [8]
3. Synchronization module

Local Storage is a client-side key-value database, meaning it is stored in the user’s browser. This means the users data is saved on their machine inside their browser. Here is example of local storage.

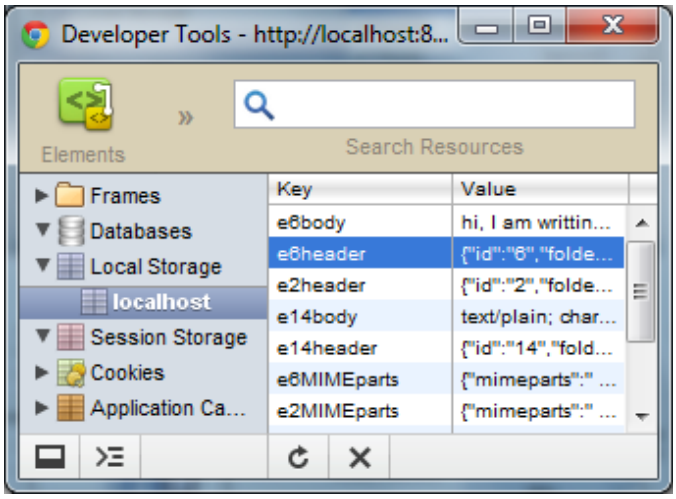

Figure 3. HTML5 local storage for emails.

2.6. Webmail Client

Webmail Client provides users with a Graphical user interface which can operate in a similar manner in both online and offline environments. HTML5, AJAX, JavaScript and CSS are the main technologies used for this module [8].

This offline web client can be divided in to two main components: the Graphical User Interface and application logic with JavaScript and AJAX. The following diagram shows the overall design of the system.

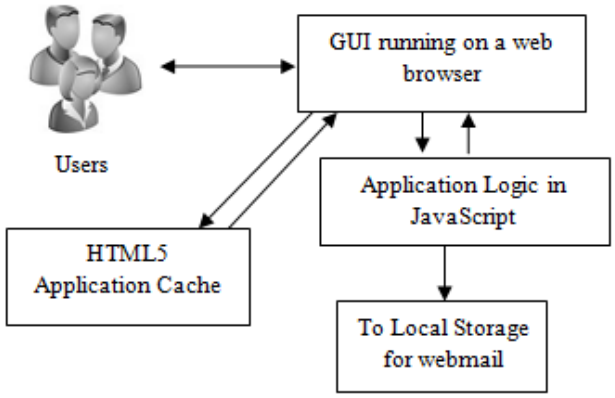

Figure 4. Components of the Offline webmail client system.

The GUI of the system is developed using HTML5 and CSS. This GUI is similar to the GUI of Horde IMP [4]. However the GUI of this system has the ability of supporting offline operations by using HTML5 application cache [10]. Since this system must have the ability to run from clients' computer, all the functionalities must be implemented by using a client side scripting language. JavaScript is the scripting language used in this system for implementing all

the functionalities. All the data required for this system is taken from the ‘local storage for webmail’ system and data to be sent to the email server are sent through the same system.

In implementing this system one of the basic but most important function is to detect the status of network to main email servers. For this purpose AJAX function is used which tries to connect to main email server time to time. Also in this system user can go to offline mode when he/she wants. When user is in offline mode or no network connection is available the system uses data stored in the local storage and stops all synchronization processes with main servers. Also all the information to be sent to main email servers are stored locally by using the ‘Local Storage for webmail’ system.

3. Results

Integration of those modules connect client offline local storage in to main server. Those email information and data that is stored browser cash indexed based database[2] in local offline mail client machines while he/she is working in offline mode.

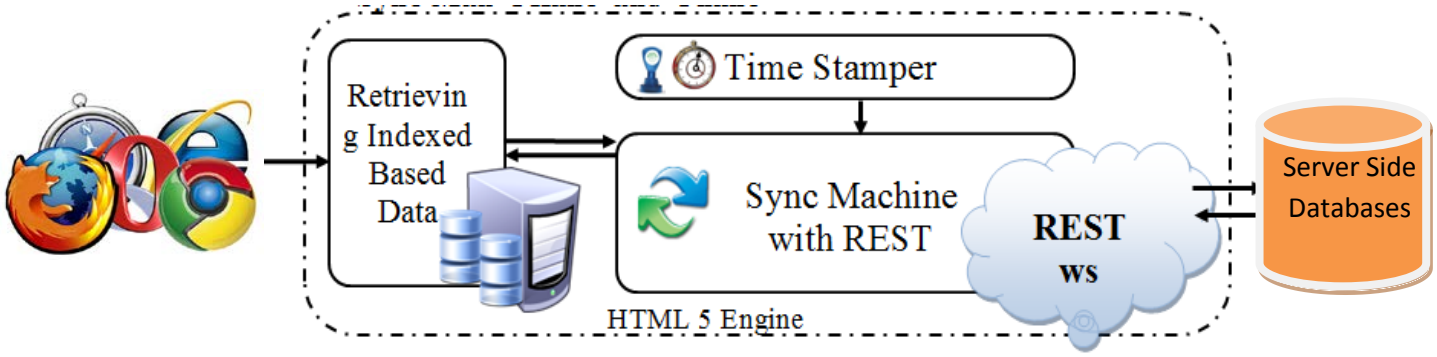

Figure 5. Top level architecture of REST offline mail.

Still his/her action or event in offline is not activated yet. When he/she go online mode, the sync mail module retrieves the data and synchronizes with the server.

4. Conclusion

If email user used email clients application such Outlook, Pegasus Mail, Mozilla Thunderbird he\she cannot get all web based email services such Google labs features. If we used normal web based email systems Gmail, Yahoo, Live etc. have no offline capabilities. Our system will store

all data event in local browser so user can use all e-mail features both while online and offline. It is world first offline web-based e-mail system.

References

1. Zoho, Google Gears, HTML5, [Online]. Available from: http://blogs.zoho.com/general/zoho-google-gears-html5.
2. Indexed Database API, W3C Working Draft 19 August 2010, [Online]. Available from: http://www.w3.org/TR/IndexedDB/#object-store-storage-steps 2010 W3C®
3. Imp Module (31/01/2010) by Chuck Hagenbuch [Online]. Available from: http://wiki.horde.org/ImpModule. ®
4. 2009 The Horde Project (2011/01/31) IMP Webmail Client and DIMP, [Online]. Available from: http://www.horde.org/imp/
5. 2011, O'Reilly Media, Inc., Creating Custom Email Queries, [Online]. Available from: http://onlamp.com/pub/a/onlamp/2004/07/22/datamining_email.html?page=2
6. "Relationship to the World Wide Web and REST Architectures". Web Services Architecture. W3C, [Online]. Available from: http://www.w3.org/TR/ws-arch/#relwwwrest
7. Pautasso, Cesare; Zimmermann, Olaf; Leymann, Frank (2008-04), "RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision", 17th International World Wide Web Conference (WWW2008) (Beijing, China), [Online]. Available from: http://www.jopera.org/docs/publications/2008/restws
8. "Offline Web Applications". World Wide Web Consortium. [Online]. Available from: http://www.w3.org/TR/html5/offline.html#offline
9. "W3C Confirms May 2011 for HTML5 Last Call, Targets 2014 for HTML5 Standard". World Wide Web Consortium. 14 February 2011. Retrieved 18 February 2011.
10. "Web Storage Specification". World Wide Web Consortium [Online]. Available from: http://dev.w3.org/html5/webstorage/
11. "Watermarking Techniques for Relational Databases: Survey, Classification and Comparison", The Journal of Universal Computer Science, vol 16(21), pp. 3164-3190, 2010.
12. Part I email formats, Wood, D (1999). Programming Internet Mail. O'Reilly. ISBN 1-56592-479-7.
13. "Registration of Mail and MIME Header Fields" from the Internet Society (2005). [Online]. Available from: http://tools.ietf.org/html/rfc4021