Computer Science Honours
Final Paper 2018

Title: A Portable Large Volume Email Retrieval System

Author: Monyemoratho Breyden

Project Abbreviation: FINDMAIL

Supervisor: Associate Professor Hussein Suleman

<table>
<thead>
<tr>
<th>Category</th>
<th>Min</th>
<th>Max</th>
<th>Chosen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement Analysis and Design</td>
<td>0</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>Theoretical Analysis</td>
<td>0</td>
<td>25</td>
<td>0</td>
</tr>
<tr>
<td>Experiment Design and Execution</td>
<td>0</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>System Development and Implementation</td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Results, Findings and Conclusion</td>
<td>0</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Aim Formulation and Background Work</td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Quality of Paper Writing and Presentation</td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Adherence to Project Proposal and Quality of Deliverables</td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Overall General Project Evaluation [requires explicit motivation from the project supervisor]</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
A Portable Large Volume Email Retrieval System

Breyden Monyemoratho
University of Cape Town, South Africa
Computer Science
mnybre002@myuct.ac.za

ABSTRACT
Email is present in all facets of daily life. A remarkable amount of information resides in email archives. This paper describes an attempt to develop a portable email system that allows users to input large email archives in various formats and accurately and efficiently browse and search over the archive offline. This approach to designing systems for preservation and offline access is useful in areas with limited Internet bandwidth such as in most African countries. Experimental results confirm that users were satisfied with the general design of the system and moreover, that this system is effective and efficient.

CCS CONCEPTS
• Information systems → Information retrieval → Specialized information retrieval • Software and its engineering → Software libraries and repositories • Software and its engineering → Software Portability

KEYWORDS
Portable; Searchable; Email formats; Archives; Indexing; Parsing; User interface; Query System

1. INTRODUCTION
Email continues to be an important and very popular form of personal or business communication, as well as a way to manage tasks and archive personal information. In the working world, there is often a need to search and browse through very large collections of emails, to track down individuals or to verify decisions, etc. For convenience, many users will either delete or archive email after it has been handled. If they choose to archive their email, these archives can later become large and cumbersome to search through, especially after a long period of time has passed. The increasing trend creates difficulties in attending to email and results in behaviours that make email feel overwhelming. The groundbreaking Whittaker and Sidner 1996 paper coined the term “email overload” to describe how disorganized emails were [15]. It is attributed to many factors, including poor personal information management and large amounts of high priority email [15].

With the problem of “email overload”, there is also the issue of archives becoming obsolete through software aging [8]. In order to combat obsolescence and improve longevity, various preservation strategies need to be considered.

A possible solution to address the email overload and obsolescence issues, is to use a portable offline searchable email archive that handles mbox and maildir email formats. The searchability feature would allow for specific emails to be retrieved from the large archive (managing email overload), while the portable and offline features would make the archive less likely to become obsolete in the short-term. This solution is the one proposed in this paper.

Taking the above into consideration, we created a Web application to facilitate portability and offline searchability, that allows for multiple email formats as inputs.

The project is divided into two logical sections which, when used in conjunction solves the overall project. The two separate sections are as follows:

1. Pre-Processing:
This involves parsing and indexing of the inputted archives of various email formats. Parsing will extract and structure relevant information from the inputted archive, while
indexing involves creating indices from the parser output.

2. Email Processing:

This involves the user interface and a query system that allows for fast and efficient retrieval of emails. The user interface should display emails clearly to the user and allow for ease-of-use. The query system should be able to handle various queries and facilitate discovery of relevant emails.

Shivaan Motilal worked on the pre-processing components and I worked on the email processing components, namely the user interface and query system. The research questions on the email processing components were as follows:

- Is it possible to create a user interface that represents emails in an easy to understand way, and is usable?
- Can a query system be created that allows for fast and accurate retrieval of email?

1.1. Project Significance

We hope this project will help individuals better manage their emails from large archives and provide them with fast and accurate search and browse functions, making it more likely for the information to be retrievable in later years.

1.3. Project Structure

The rest of this paper presents background work and how the findmail system was designed. Experimental design and various experimental results are then discussed to illustrate how findmail is effective and efficient. Finally, ethical considerations, conclusions and future work are presented.

2. BACKGROUND

2.1. Digital Collections:

For developed countries, many preservation techniques can be implemented, however this is not the case for developing countries (such as in Africa) [12]. In developing countries, most preservation techniques cannot be implemented due to insufficient resources and poor/expensive cost of Internet bandwidth.

A particular way of preserving digital collections (including email archives) that works for developing countries is through using the principle of simplicity [8]. An illustration of this could be the use of XML plain text documents to store information and metadata, making it more likely for the information to be retrievable in later years. Focusing on simplicity also provides easier interconnection, extension and modification of the features of a specific system, allowing for the system to function on multiple platforms (portable) [8]. The concept of portability is important for email, as email users use multiple platforms to access their email, and the email itself can be stored in different formats.

Suleman et al. [13] developed CALJAX, a generic hybrid (online-offline) repository management and access system based on a strong AJAX foundation. It allows integration of content from a local source with content from a remote source, with the only requirement being a Web browser. XML plain text documents were used to store information, making it more likely for the information to be portable, preservable and accessible through a Web browser.

Expanding on the issue of poor Internet bandwidth, is the idea of having hybrid online-offline digital collections to counteract this issue. Online and offline collections present both advantages and disadvantages, thus a hybrid digital collection (online-offline repository) could interleave advantages from both, and potentially aid in preservation [13]. A hybrid system was however not in the scope of this project.

2.2. Email Archives:

Some existing software projects around email archives include Windows Mbox Viewer (MV) [11], Mairix [10] and Mailpile [6]. WMV [11] displays mbox files on the user’s screen via a simple user interface. It runs offline but is a program specifically for Windows. It also does not provide search functionality over the archive. The other downsides are the fact that it does not cater for other email formats and is not portable across operating systems.

Mairix and Mailpile include indexing and search functionality, but are not suitable either in terms of preservation, portability or offline use. Mairix [10] is an email indexing and searching tool that works with maildir, MH or mbox formats. It works offline but is
mainly for Linux systems. Since it involves installation and is not portable across non-Linux operating systems, it is unusable in this project.

Mailpile [6] is similar to Mairix; it also indexes mbox and maildir formats, however Mailpile is an email client and personal Web mail server. It also has a much better user interface (in comparison to WMV) that is based on Gmail. It works on multiple browsers but does not have specific offline usability. It was made using Python, JS and HTML5, and is the closest work to the one we propose in this paper.

3. DESIGN OF EMAIL PROCESSING COMPONENTS

The email processing is split into two components namely, the user interface and a query system.

3.1. User interface(UI)

The access Web interface is a standard email Web interface offering the user search and browse functions. It was developed using a user-driven approach in order to understand users’ needs and preferences. It consists of mainly static HTML, CSS and Javascript to display the relevant result when a user invokes one of the services. The indices are accessible to the web browser, and can be parsed using Javascript. This pre-indexing process is slow compared to the actual search, but is necessary to obtain fast search results. For browsing, applicable pre-generated indices are parsed using Javascript and displayed to the user. For search, returned indices are also parsed using Javascript and displayed to the user.

3.1. Query

The query system, using extended boolean implementation retrieves relevant emails from the email archive, by using the indices generated by the search indexer. An index file either matches the query or it does not. This provides greater control and transparency over what is retrieved. Within an index file, the listing of email document together with the query term frequency occurs (the number of times a term occurs in the email document). The search algorithm retrieves a set of matching documents ranked by the number of times a term occurs in the email document. The returned results (indices) are then parsed using Javascript and displayed to the user.

![Fig. 1. Overview of the FINDMAIL system](image)

Figure 1 shows the high-level architecture of the system. The popular email formats: maildir and mbox, are inputted to the parser to extract relevant information. The parser then sends its output to the two indexers. The browser indexer will then create indices to facilitate browsing of the email, while the search indexer will create indices for the search functionality. Both of these indexers will interact will the user interface to provide the services of browsing and searching to the user.

4. EXPERIMENTAL DESIGN

In order to test the usability, performance, portability and relevance of the search results, it was necessary to develop a set of experiments below divided into logical sections.

4.1. System Usability Testing:

The usability test was conducted on a near final version of the software. Second year computer science students were recruited through a convenience sampling method (as the experiments required basic computer literacy skills). The study was advertised via email. The test was conducted with a total of 20 students to assess attributes of the system that make it understandable, learnable, easy-to-use and attractive. The task scenarios were designed to assess compliance with recognized usability principles [3]. The exact tasks of the evaluation can be found in Appendix A. The data was sourced from our own personal Gmail inboxes. The data was of a high quality and it provided a representative sample of the inputs that are likely to be
Participants were asked to sign a consent form informing them of the anonymity of their results. On completion of the task scenarios, users were asked to fill in a system usability score questionnaire to determine the usability of the system. On completion of the usability study, users were compensated for their time with a standard fee as specified by the Department. Tests were conducted in an uncontrolled environment of the Computer Science Senior laboratory.

Participants accessed a Web page (standard email page) that presents a browse view of the collection, using a laboratory computer through Chrome browser. Rather than observe users throughout the test process, users were allowed to conduct tasks and answer questions independently within the 20 minutes of the usability test session. The reason for this is that users who are observed will alter their behaviour and may become nervous, resulting in mistakes and errors affecting results. However, if users experienced particular difficulties in completing a task or found the instructions to be ambiguous, the facilitator could be asked for help or clarification. Responses were constrained to a Likert scale that ranges from ‘strongly disagree’ to ‘strongly agree’.

4.2. Performance Testing:

Experiments were conducted to measure the time it takes to download and display the entire content of a Web page for both the search and browse functions over collections of various sizes. The data collection used were simple text files filled with test data. This allowed tight control over the number of files, as the exact number of files could be generated for each test. The test was conducted on collections containing various number of files (2000, 4000, 6000, 8000, 10000, 12000) and all the files contained the same email items, as the load time is affected by the number of HTTP connections needed to download items, item size and types. The browsing test was conducted by loading 3 pages and the load times were recorded and averaged. The searching test was conducted by searching for sampled query term(s) present in the collection and also averaging the time to generate the results view (check Table 1). The load times were measured using the performance.now() utility coded within the system. All performance testing was done on a Mac Book Pro 5.2.

4.3. Portability Testing:

Cross browsing tests were conducted to study whether the look and feel as well as functional features of the developed system worked as intended across popular browsers (Microsoft Edge, Google Chrome, Mozilla Firefox and Apple's Safari). This involved studying the following metrics:

- User Interface: Checking to make sure the UI matches the original plans.
- Behaviour: Checking to make sure functional features throughout are the same.
- Code validation: Checking to make sure Javascript and CSS validates across the different browsers.

4.4. Relevance of the search results:

To measure the relevance of the search results, the test was conducted using our personal collections containing 10 files. For certain sampled query term(s) present within the collection, precision and recall were measured.

5. EXPERIMENTAL RESULTS

5.1. System Usability Testing:

The raw data and mode data from the usability test is provided in Appendix B. Users wanted clearly defined visuals and graphics. This was apparent when they struggled to identify chained mails, wanted a button near the search area and failed to identify emails with attachments. Thus, more information should be added for ease of understanding. There is no status shown while the searching is going on and suddenly the results appear. Users were uncertain about the search they made. This violates heuristic that user should always be notified about the things happening in the system [3]. When looking at design, all the users were happy with the basic and minimalistic design. A majority of users (>52%) believed the system was not lacking in intuitivity. The system had maximum cognitive flow (little friction and confusion when the user was using the system).
The overall feedback was positive, with all criticism being constructive and leading to consistent improvements and updates to the design of the user interface.

5.2 Performance Testing:

The results of the time taken to generate a browse view are presented in Figure 2. This time is roughly linear with increases in collection size. Similar results were obtained for the search function, as shown in Figure 3.

There are a few caveats that we were aware of for this kind of performance measurement as listed below:

1. The available system memory and CPU.
2. The browser used affected the Javascript execution and rendering speed.

Figure 2. Time taken to browse a collection.

Figure 3. Time taken to search a collection.

5.3 Relevance of the search results:

Table 1 shows that the system returns approximately all the relevant result sets.

<table>
<thead>
<tr>
<th>Query/Term</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>gary</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>projects</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>technical</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Research</td>
<td>1</td>
<td>2/3</td>
</tr>
<tr>
<td>participant</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

5.4 Portability Testing:

The figures below show screenshots of the User interface across different browsers. The user interfaces across the different browsers, worked as intended with respect to the look and feel, code validation and functional behaviour. For this reason, the developed system can run on any of the popular browsers without any change in behaviour and look and feel.
6. ETHICAL, PROFESSIONAL AND LEGAL ISSUES

Ethical issues were identified in the testing, software implementation and data handling stages of the project. Each will be discussed in further detail below.

6.1. Testing:

We applied to the Faculty of Science Research Ethics Committee for ethical clearance, in order to test the usability of the system with students. All user testing was conducted through simple surveys and usability testing, which did not raise any ethical issues.

6.2. Software:

This project is declared open source. This is to encourage further development and improvement to our software.

6.3. Data:

We sourced data from our own personal Gmail inboxes (Shivaan Motilal has a 680 MB inbox unzipped, Breyden Monyemoratho has a 670 MB inbox unzipped) and compiled from the Enron email dataset containing approximately 1.5 million emails (423MB, tarred and zipped) [2], which is freely available for reuse.

7. CONCLUSIONS AND FUTURE WORK

Experiments have confirmed that the developed proof of concept is intuitive, portable and effective for
browsing and searching over email archives. Findmail has demonstrated that it is possible to leverage a simpler architecture and Web technology to enable fast and accurate browsing and searching over email archives in developing countries with limited Internet bandwidth.

This simple approach can be extended further in the following ways:

1. Leveraging the feature of the AJAX framework to enable integration of content from a local source with content from a remote source, thus allowing the user full access to the most current content.
2. Integrating the current solution with tools and services that facilitate preservation, such as logging and integrity checking.
3. For greater efficiency, splitting browsing and search indices into shards. Thus the speed of both operations will be constant irrespective of the size of the collection.
4. Some (advertising) email messages contain unsightly links that are bundled together with the message body. There is no easy way to ascertain if these links will be useful to the user or not, however the display of these links can be improved on.
5. The display of threaded email to the user can be improved on. Threaded messages are currently separated into parts but not displayed as such to the user.
6. Using Dublin Core as the metadata scheme to ensure conformance to international standards and a universal understanding of the metadata.

7. ACKNOWLEDGEMENTS

I would like to thank my project partner Motilal Shivaan and project supervisor, Associate Professor Hussein Suleman for his commitment and help throughout the course of this project. Finally, my sincere thanks and appreciation are extended to University of Cape Town’s Computer Science department for funding our participant remuneration and supplying experimental space.

8. REFERENCES

computing systems (pp. 276-283). ACM. April, 1996.
APPENDIX A: User Interface task scenarios:

1. Check all buttons and directories are properly labeled and functional.
2. Verify that on clicking the email, user is navigated to email content.
3. Check availability of all the mandatory fields like: sender, subject, body and attachments etc.
4. Verify that any attachments can be opened and are downloadable.
5. Check chained mail display.
6. Enter a single word query to search in the search-box. eg. your name: “Tom”
7. Enter a sentence or phrase query to search. eg. “Science faculty”

APPENDIX B: The raw data and mode data from the usability test: