

Eudora2 Mailstore organization on disk

In Eudora2, we introduce an entirely new way of storing mail messages on disk.

Rationale

We will do this for several reasons:

- We want to keep the entire message structure. Currently, Eudora throws away much of the message structure when the message arrives. This makes checking digital signatures, to give just example, very difficult.
- Cross-platform compatibility. We expect that users will be able to move their Eudora2 mailstore from a Macintosh system to a Windows system (and back) and use it without change on either one.
- Spotlight support. In Mac OS X 10.4, Apple introduced Spotlight, a service for indexing data files and doing user-level fast searches. Unfortunately for us, Spotlight only works on individual files - it has no concept of "containers", files that contain multiple addressable items (like a database).
- We want to make the mail system less susceptible to corruption. Currently, if you have an error in a mailbox, you could potentially lose every message in the mailbox.
- We want to support search-based mailboxes, and also the more general case, where a message can reside in more than one mailbox.
- We want to get rid of the attachments folder. Splitting out the attachments was a good idea back in the early 1990s, when disks were small, and processors were slow. Keeping the entire message will enable us to leave malicious attachments in non-viable form by default (base64 encoded), at a cost of some disk space.
- In Eudora, putting a single message into a 10,000 message mailbox means that you have to back up 10,001 messages. Also, Eudora's behavior (open a mailbox, append a message, close the mailbox) leads directly to file fragmentation. We'd like to keep backups small, and files contiguous whenever possible.

Structure

- Each message will be stored in its entirety on disk in an individual file, along with additional information. (Such as Read/Unread/Replied status, a list of the mime-parts, and so on).
- Each mailbox will consist of a list of "pointers" to messages, plus a bit of additional information. (Some information that is stored in the messages will be cached, for performance reasons, in the mailbox).
- Each message will contain a list of mailboxes that it is "contained in", and when a message is not in any mailboxes, it will be deleted from the mailstore. (Reference counting)
- Some mailboxes will be "ad-hoc", i.e. will just be collections of messages. Others will be "Search-based", and will consist of the results of search criteria, and can change as the underlying messages change. For example, users may want to keep a "UnRead" mailbox, which contains all the mail that has not yet been dealt with. As a message is read, it will be removed from the "UnRead" mailbox without any user action (other than reading the message).

Details

- *Collection.* A data structure that is used all over the Eudora2 mailstore is a "collection", which is merely a list of untyped data, indexed by 'type' and 'id'. This is very similar, conceptually, to an old-style MacOS resource file, except that the data is kept in memory, not on disk. A collection can be converted to XML, which can be saved to disk. Both messages and mailboxes contain a collection, which is an extendable data structure that can be used by all parts of the mailstore and its' clients.
- *Hash-ID.* Several data types in the Eudora2 mailstore are identified by 32-bit identifiers, which are generated from some text value. These are called Hash-IDs.

Organization on disk

- *Mailstore.* A mailstore will consist of two folders, one named "Messages" and the other named "Mailboxes", along with a file named "Mailstore.xml". [!!! This file should have it's own suffix like "EuMStore"] The XML file will contain:
 - The collection for the mailstore.
 - A list of messages with pending updates.

The *mailboxes folder* will contain the mailboxes., and each mailbox file will be named "<Hash-ID>.euMBox", where Hash-ID is the hash of the original name of the mailbox (effectively, some random number). In the case of a hash collision (trying to create a mailbox with a hash-id that already exists, another hash-id will be chosen.. There is no sub-structure to the mailboxes folder.

The *messages folder* will contain the mailboxes., and each message file will be named "<Hash-ID>.euMsg", where Hash-ID is the hash of the "Message-ID" header contained in the message. In the case of a hash collision, either due to messages with the same Message-ID or just a collision in the hash function, a folder named "<Hash-ID>" will be created, and files with the names "<Hash-ID>.<sequence#>" will be created inside that folder. At present, there is no sub-structure in the messages folder, but it is expected that there will be one at some point, in order to keep the file system performance reasonable. (we don't want a folder with 300,000 files in it).

- *Message files.* A message file has the suffix 'euMsg', and consists of a header, the message, and a footer. The header is a single line consisting of the characters "%E2-001<space>", followed by the size of the message (in bytes), followed by <CR><LF>. The message is the entire message (headers, mime parts, separators, etc, exactly as it came off the wire. Each line of the message is terminated by <CR><LF> (again, just like it came off the wire). The footer contains an XML fragment containing at least:
 - The Message-ID for this message (Hash-ID + sequence number)
 - The collection for this message (which contains several entries)
 - The list of mailboxes that this message "belongs to"

This means that the entire message can be read as a <CR><LF> delimited text file.

- *Mailbox files.* A mailbox file has the suffix 'euMBox', and consists solely of XML. The XML contains:
 - The Mailbox-ID (and the ID of the mailbox's parent)
 - The collection for this message
 - A list of messages that are contained in this mailbox.