Symig

A more unixy way of dealing with mail, in Git

Runxi Yu
`https://runxiyu.org`

Lindenii Project
`https://lindenii.org`

2026-01-30

- Mail user agents, email clients, etc., are a bit of a pain.

- Mail user agents, email clients, etc., are a bit of a pain.
- Message are organized into folders.

- Mail user agents, email clients, etc., are a bit of a pain.
- Message are organized into folders.
- Messages are opaque-ish objects, with IMAP flags.

What if messages are actually files?

What if messages are actually files?

That you could just `mv` around, `rm`, etc?

What if messages are actually files?

That you could just `mv` around, `rm`, etc?

Over a synchronization primitive that robustly handles conflicts?

Git

Repository structure

Almost what you want it to be. . .

```
archive/
 1970-01-01_00:00:00_jane.doe@example.org_Hello_nonce.eml
drafts/
sent/
symig.deliver
symig.indexing
symig.sieve
1971-01-01_00:00:00_jane.doe@example.org_Hello_nonce.eml
1972-01-01_00:00:00_jane.doe@example.org_Hello_nonce.eml
```

Mail repository delivery

- Each recipient resolves to a Git repository.

Mail repository delivery

- Each recipient resolves to a Git repository.
- The daemon fully resolves `HEAD` to a detached commit.

Mail repository delivery

- Each recipient resolves to a Git repository.
- The daemon fully resolves `HEAD` to a detached commit.
- All delivery work happens against that commit snapshot.

Filtering and placement

- Read `symig.deliver` for delivery configuration.

Filtering and placement

- Read `symig.deliver` for delivery configuration.
- Run the sieve script in `symig.sieve`.

Filtering and placement

- Read `symig.deliver` for delivery configuration.
- Run the sieve script in `symig.sieve`.
- Script decides message format and target path (e.g. `inbox/`, `archive/`, etc).

Committing mail

- A new tree and commit are created with the message added.

Committing mail

- A new tree and commit are created with the message added.
- The daemon updates the ref with CAS semantics.

Committing mail

- A new tree and commit are created with the message added.
- The daemon updates the ref with CAS semantics.
- On conflict, retry on the new ref head; no rebases, no merges.

Symig daemon

- Accept Internet Mail over LMTP/SMTP
- Use a resolve table to recursively expand aliases and resolve recipients
- For each recipient, deliver to its corresponding Git repository.

Client utilities

symig-unpack	unpacks selected parts from a MIME message
symig-header	reads headers from a MIME message
symig-index	indexes a mail repository
symig-find	finds emails from the index
symig-find-unpack	finds an email from the index and unpacks
symig-find-header	finds an email from the index and reads headers
symig-attach	attaches an attachment to a MIME message, converting the MIME message into a multi-part MIME message if necessary
symig-threads	uses the index to find emails in the same thread as the provided one.
symig-compose	create a new message based on a template, optionally adding `References` and `In-Reply-To` headers from the `Message-ID` of an existing message

Only the basics implemented at the moment but growing!

codeberg.org/lindenii/symig