Ryan McElroy
Jason Salameh
CSE 403

# CSE 403 Proposal LCO

## 1. Operational Concepts:

- **Overview:**

  Update Mailman mailing list software to version 3.0 by the addition of several major features, namely:
  - Relational Database backend (MySQL or SQLite?)
  - User-centricity (currently Mailman is list-centric)
  - Flexible, extensible, skinnable web archiver
  - Flexible, extensible, skinnable web interface

  In more or less that priority.

  In addition, features can be drawn from the Mailman developers to-do list: <http://www.gnu.org/software/mailman/todo.html>

- **Impact:**

  Tens of thousands of companies, organizations, and internet sites use Mailman mailing lists, and all Mailman installtions, unless heavily modified, are subject to the basic feature limitations inherent to the original project. This long-due update to Mailman would have a vast worldwide impact on a wide variety of companies and organizations, including all UW mailing lists and mailing lists for companies such as Apple and Dell.

- **Notes:**

  Development will occur in Python, a relatively new, interesting, and popular language.

## 2. System Requirements

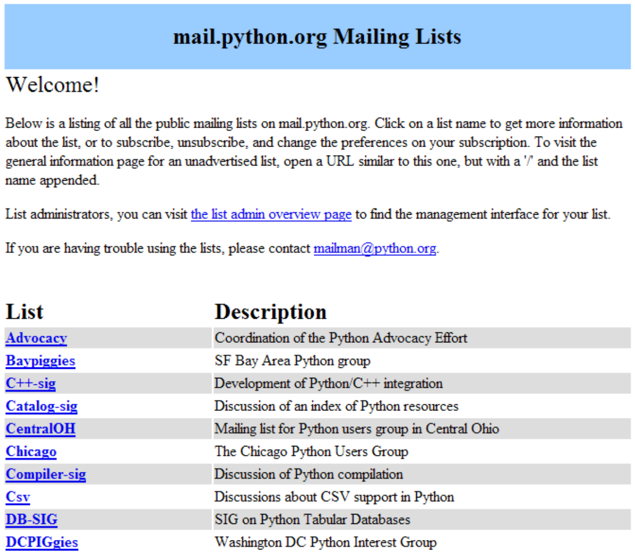
A user of the updated Mailman software will have a considerably different experience than a user of the current software. Currently, the software is list-centric, meaning that each list has its own page where a particular user can sign up an email address, and a user must sign up separately for each list on the site. Furthermore, to send email from another account, the user must also sign up the second account, leading to perhaps unwanted duplicate emails. The new, user-centric model would emphasize user accounts that allow multiple email addresses and one-click sign up for multiple lists on the same server. A user could have different lists send emails to different email accounts, but accept email from any account.

A user will also see the all-too-familiar "Mailman look" go away in favor of much more tightly integrated Mailman web sites (Even Apple's installation is clearly identifiable). The current "slightly flexible" architecture of the web front end will be replaced with a much more flexible and extensible module that allows tight integration into existing websites.

An administrator of the updated Mailman software will also have a considerably different experience. By using a relational database as the backend of the system, there will be a variety of administrative improvements:

- Single-point changes with guaranteed system-wide visibility (in current versions some information is duplicated in many places)
- Instant style updates (currently, lengthy archive "rebuilding" is required)
- Powerful searching, editing, and other administrative features that are currently prohibitive due to flat-file data storage

Web designers will also be able to finally really customize the look and feel of Mailman web sites, due to the addition of a much more flexible web archiver and web interface. Styling will be updated to use modern standards-compliant HTML and CSS. The default look (below, left) will be easily updated to a custom look (below, right), or even better than that.



**mail.python.org Mailing Lists**

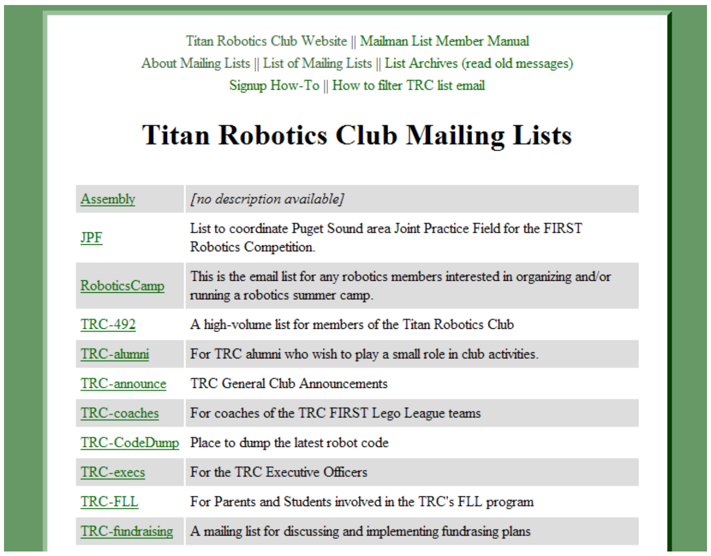
Welcome!

Below is a listing of all the public mailing lists on mail.python.org. Click on a list name to get more information about the list, or to subscribe, unsubscribe, and change the preferences on your subscription. To visit the general information page for an unadvertised list, open a URL similar to this one, but with a '/' and the list name appended.

List administrators, you can visit the list admin overview page to find the management interface for your list.

If you are having trouble using the lists, please contact mailman@python.org.

| List | Description |
|---|---|
| Advocacy | Coordination of the Python Advocacy Effort |
| Baypiggies | SF Bay Area Python group |
| C++-sig | Development of Python/C++ integration |
| Catalog-sig | Discussion of an index of Python resources |
| CentralOH | Mailing list for Python users group in Central Ohio |
| Chicago | The Chicago Python Users Group |
| Compiler-sig | Discussion of Python compilation |
| Csv | Discussions about CSV support in Python |
| DB-SIG | SIG on Python Tabular Databases |
| DCPIGgies | Washington DC Python Interest Group |



Titan Robotics Club Website || Mailman List Member Manual
About Mailing Lists || List of Mailing Lists || List Archives (read old messages)
Signup How-To || How to filter TRC list email

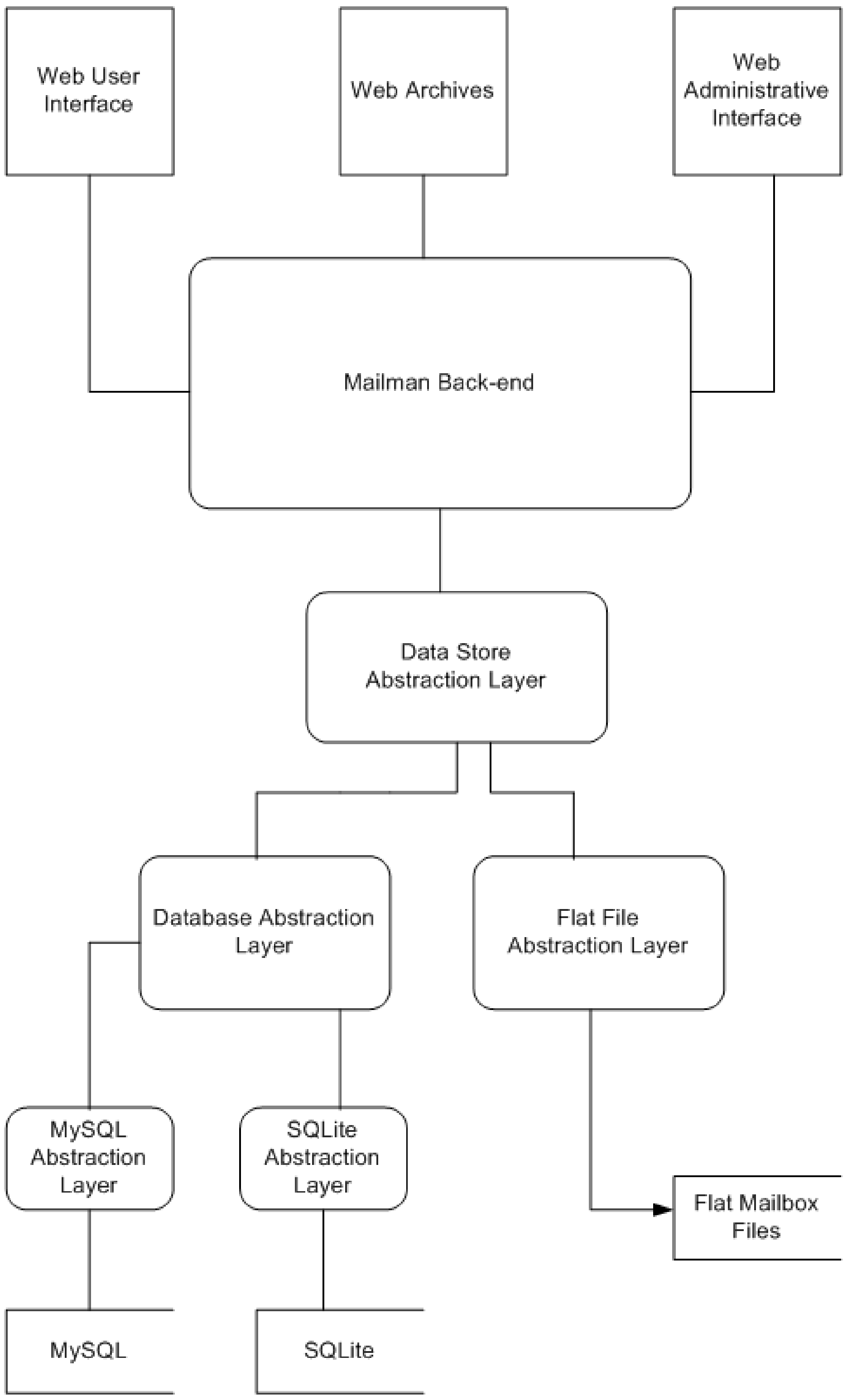
**Titan Robotics Club Mailing Lists**

| | |
|---|---|
| Assembly | *[no description available]* |
| JPF | List to coordinate Puget Sound area Joint Practice Field for the FIRST Robotics Competition. |
| RoboticsCamp | This is the email list for any robotics members interested in organizing and/or running a robotics summer camp. |
| TRC-492 | A high-volume list for members of the Titan Robotics Club |
| TRC-alumni | For TRC alumni who wish to play a small role in club activities. |
| TRC-announce | TRC General Club Announcements |
| TRC-coaches | For coaches of the TRC FIRST Lego League teams |
| TRC-CodeDump | Place to dump the latest robot code |
| TRC-execs | For the TRC Executive Officers |
| TRC-FLL | For Parents and Students involved in the TRC's FLL program |
| TRC-fundraising | A mailing list for discussing and implementing fundrasing plans |

## 3. System and Software Architecture

The major task is the integration of a relational database as the back-end of the Mailman software. The current flat-file system will likely be maintained as an option, with the new default being the relational database. This will require the addition of several abstracting layers -- first, a layer that abstracts the infromation that the other parts of the code want from the method retrieving that information (be it flat-file or database). Next, a database abstraction layer will allow the use of several different database packages with the program. Finally, one or two database drivers will interface between the database abstraction layer and a specific database, such as MySQL or SQLite.

The programming language will primarily be Python, with perhaps some PHP for the website front-end integration (this is to be determined). The database to be used will be either MySQL or SQLite. Drivers may be written for both.



## 4. Lifecycle Plan

We may use the Extreme Programming model to ensure high-quality, releasable code throughout production

Major Milestones:

2007-06-08 -- Final Project Done
2007-05-25 -- Third Release -- Feature Complete, Integrate & Debug Only

2007-05-11 -- Second Release -- Major features implemented
2007-04-27 -- First Release -- Database back-end completed
2007-04-13 -- Architecture complete -- Interfaces and contracts defined

Ideally this project will include 8-12 engineers, divided into front-end and back-end groups. The back-end group should have experience with relational database application design, while the front-end group should have a bit of creative flair and web application development experience.

## 5. Feasibility Rationale

This is an ambitious project with many large risks. However, the risks are manageable because the project is very scalable. For example, if it is determined that implementing a database back-end will take longer than expected, there is an extensive todo list (see first section) where other reasonable enhancements can be chosen from. One key assumption is that a database module will be able to be developed without a complete rewrite of the Mailman codebase. If this assumption is false, then the nature of the project may have to be revisited.