FREEFOUNDRY

General plan

Railcoach project

Joost Elfering

9/6/2010

This documents describes the railcoach applications idea, philosophy, features and planning. The railcoach project will provide the technical implementation of the [openfoundry] project platform.

ABSTRACT

//TODO

GOAL

The goal of the project is to create a technical implementation for the [openfoundry] project platform. The [openfoundry] is to be a platform for projects with a technical basis. The railcoach project will provide the technical implementation for the [openfoundry] as a platform.

The goal of the platform is to provide the place, tools and social environment for projects. The place is the website itself and the organizational structure it represents. The tools can be anything but will generally assist in collaboration, documentation, announcing and deployment for a project. Social environment is created by the user profiles and the connections between them.

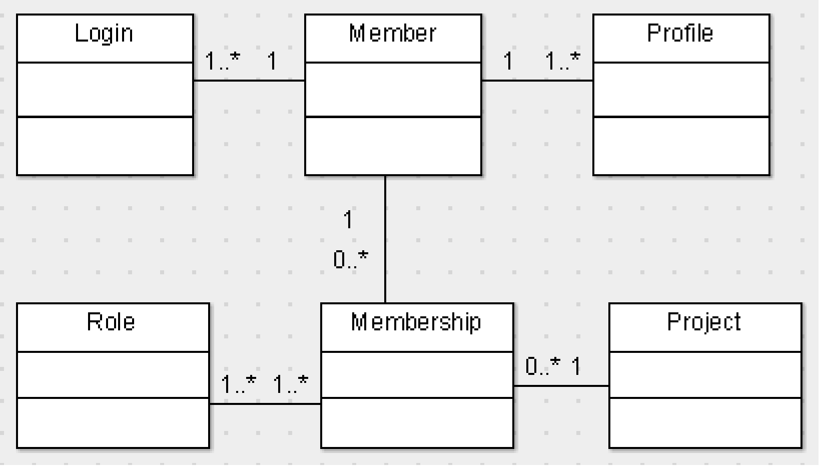
Our goal is not to replace a place like github, Google code, rubyforge or sourceforge. The organizational background is different and the technical platform will need to represent that difference in background.

FEATURES

From the goals we can derive the basic features and some hint of the more detailed features. We will need project groups, users with profiles, memberships, friends and the tools that they need to run the project.

BASIC STRUCTURE

The data structure has already been on the table. Below you will see a representation of the technical model. The users and projects are well represented in this model, interpersonal relations are not represented.



As you can see above an authentication adapter allows the members to log into the system. Members have one or more profiles to represent them. Users can become member of a group and a user can have a couple of roles within a project. The project holds the rest of the functionality.

FEATURE MAPPING

Creating a mapping of the application can give us an insight into all the features we need and how they integrate into the basic model of the application. This list can also be used as an overview of the application.

- Users
    - Profile(s)
        - Personal info
        - Showcasing
            - Projects
            - Contributions
            - Roles
            - Activities
        - Work history
    - Authentication
        - Open-id
        - Facebook connect
        - Native authentication
    - Personal Tools
        - Version control repositories
            - Viewer
            - Manager
        - Blog
        - Agenda
- Project
    - Profile
        - Promotional article
        - Activities
    - Tools
        - Announcing
            - Blog
            - Releases
        - Collaborating
            - Forum/mailing lists
            - Version control repositories
            - Issue tracking
            - Continuous integration
        - Documentation
            - Wiki
            - CMS
        - Deployment
            - Build servers
            - Hosting
            - Release packaging
        - Sub-projects
        - Agenda

- General showcasing
    - Blog planet
    - Member featuring
    - CMS promotion/showcasing
    - Releases
    - Upcoming events
    - General activity stream

FEATURE LIST

Profile:	project/promotional/user
Blog:	project/promotional/user
Static content (pages):	project/promotional/user
Version control:	project/user
Activities:	project/promotional/user
Agenda:	project/promotional/user
Forum:	project
Issue tracking:	project
Continuous integration:	project
Build servers:	project
Issue tracking:	project
Releases:	project/promotional
Hosting:	project

FEATURES UNDER DEBATE

There are some features that are still in limbo. How do we provide 1 good tool for documentation and collaboration? What do we do with static pages? How can we document discussions? Therefore the next features are still in limbo:

- Discussion tools:
    - Forum
    - Mailing lists
    - Wave
- Documentation tools:
    - Wiki
    - CMS
    - Version control

GENERAL FEATURE REQUIREMENTS

We have a couple of fairly strict requirements for any feature we will be creating. These represent our goals and ideals.

- The use of open standards and open formats. When not writing it in a natively open format or standard a secondary source will need to be supplied in an open format or standard.
- The effort to get your things into the system need to be minimal. Integration with existing networks like facebook, twitter, linkedIn and other are a goal, not only a tool.
- //ADD MORE

ROADMAP

1. Create basic functionality for users, authentication, basic user data and basic profile(s).
2. Add basic functionality for projects, basic profile(s) and information page(s).
3. Blog system basic functionality for projects and users.
4. Add basic tool integration for projects and users. Version control, issue tracking and others.
5. Static content pages basic functionality for general use and projects.
6. Events basic functionality for projects and users.
7. Enhancements/Integration.

The priorities are setup so the basics are up first. With this basic functionality in place it can be expanded and improved upon. This is in sync with the agile nature we have. Create a working application as fast as possible and improve the application as needs accumulate for specific features.

EFFORTS

All this can come down to a couple of efforts to build up the basic application that we can use. Initial integration can be crude. A link with instructions to a repository for version control is good enough at first. Not everything has to be as cool as it can be the first time. Just keep in mind that some things still need to be build upon the basic structure.

- Users
- Projects
- Blogs
- Tools
- CMS
- Events

PROJECT TOOL REQUIREMENTS

- Source code version control.
  - Preferring GIT
  - Optional could be suggested
- Issue tracker.
  - Preferring redmine
  - Optional could be suggested
- Continuous integration
  - Preferred Hudson
  - Optional could be suggested
- Documentation
  - Optional RDOC
  - Optional Wiki
  - Suggestions welcome