Workplan2011

Version 41, last updated by meijer at 2011-06-10

How to read this workplan

FileSender is an agile project. This means our workplan is by no means written in stone but is adapted as circumstances warrant and opportunities allow. There is a very good idea on where we want to get but the road there is not always clear and without bumps. This affects release planning and that is reflected in work plan changes.

Introduction

Removing the Google Gears dependency for file uploads larger then 2GB has highest priority after finalising the FileSender 1.0 release. In addition we want to put FileSender on the path to full removal of the Flash dependency. The way to go is the HTML5 FileAPI for supporting large file uploads and move towards a native HTML UI without Flash. This might need a phased approach.

In addition the "why not use a web application framework" question came up several times in 2010. This year we want to investigate whether it is worthwhile to move from "raw" PHP to a web application framework like for example Lift or Django. Having completed FileSender 1.0 offers us a good opportunity to consider such a move.

The work plan is divided in 4 rough chunks:

- finish Filesender 1.0 release
- make decision on technology platform for next 2 years
- develop and release FileSender version without Gears but with >2GB file upload support, feature-parity with FileSender 1.0
- develop other features according to priority

Rough release planning

With each release a certain amount of quality assessment, testing, documenting is required. We also know that when releasing a beta version for field testing, it starts a cycle of beta_release-deploy-test-feedback-code-new_beta_release that takes at least 6 weeks to complete. Taking these two boundary conditions into account it means we aim for 2 proper releases in 2011, and are happy if we can have smaller incrementals of good quality.

- before EU summer holiday 2011: FileSender 1.5 release with HTML5 frontend
- November: Next FileSender release
- interim releases as appropriate

Project team

The core development team consists of:

- Chris Richter: support for 1.0 release, lead developer front end, 1.5 back end, developer
- Maarten Koopmans: lead architect post-1.5 development, developer
- Xander Jansen: chief release management, testing
- Wendy Mason: chief documentation, testing
- Guido Aben: vice-project lead
- Jan Meijer: project lead

Consortium representatives

FileSender development is funded by a consortium of NRENs. Funding primarily pays to hire our two developers: Chris Richter and Maarten Koopmans. Consortium funding members and their representatives for 2012 are:

- AARNet, Guido Aben
- Belnet, Mario Vandaele
- HEANet, Brian Boyle
- SURFnet, Rogier Spoor
- UNINETT, Jan Meijer

Work break down

1. Finalise 1.0 release (*1 January - 31 January*):

- complete documentation, clean up existing documentation
- re-order config.php
- polish packages
- QA packages

2. Support 1.0 release until 1.0 becomes unsupported (31 January - max. 31 December 2011)

- provide security fixes as needed
- provide small bug fixes as needed

3. Preparation phase for next release development (*1 January - 31 March*)

- Assess desirability of using an integrated web application framework
- Prototype uploads with HTML5 FileAPI in web application framework
- Prototype uploads with HTML5 FileAPI against current backend
- Decide path to follow: evolution of current backend, or replace current backend with integrated web application framework (e.g. Lift)

Note: at the time of updating this work plan, the decision was made to proceed in an evolutionary way and build on the 1.0 code base, using existing test, packaging and deployment procedures. Development will use existing PHP libraries where possible.

Guiding principles:

- rough consensus, working code
- user experience is fire-and-forget
- less is more: design the UI for people with better things to do with their time. Less clicks is better, less clutter is better
- aim for a UI that is as clean as possible
- aim for a user experience that closely follows the email paradigm

4. FileSender 1.5: New front-end UI on existing 1.0 backend code base

- Remove Gears dependency, remove as much of Flash as possible
- Aim for HTML+JavaScript only front end
- Feature-parity with FileSender 1.0: no new features, 100% focus on removing Gears and as much of Flash as possible
- Use HTML5 FileAPI for file upload of any size, currently supported by at least FF4 and Chrome, expected to become future standard for all browsers
- Ensure reliable progress bar for uploads <= 2GB in non-HTML5 FileAPI browsers (FF3.x, IE7,8,9)

- HTML-only admin interface
- Backend changes where needed
- Opportunistic backend code cleaning (clean it when you see it) to remove redundant pieces
- Code security audit
- to be released as FileSender 1.5
- make sure FileSender 1.5 works with SimpleSAMLphp 1.7.0+ (ticket #271)

5. FileSender 1.6: backend improvements

- The Quick Win release
- **Database abastraction.** Implement and test for Postgresql and MySQL. Implement database schemas, indexes/optimalisations and test thoroughly. [READY for 1.5]
- **New database schema** to provide foundation for other features, with indexes and optimisations. Needs to allow for pause/resume/selective resend which require database to know which chunks already have arrived, sending multiple files in one go, file integrity checksums, multi-file-upload, quota support, multi-tenant, REST-API.
- **Revisit email bounce handling and dealing with SPF** (ticket #384) [READY for 1.0.1]

- **Localisation 1**: enable use of multiple language files. UI must support the user selecting different languages (with flag or text) [READY for 1.5]
- **Localisation 2:** use browsers built-in language preference to auto-select preferred language (wrap up what doesn't work in 1.5)

FileSender 2.0: enabling backend for new features

Systematic code modularisation

REST-API

- 1. adapt database schema to support improved file upload handling (multiple files in one go, resuming): database schema in such a way that production services can easy migrate: users get something new, don't need to migrate?
- then API
- upload client on ipad/iphone? (low prio)
- question: which features conflict with Flash component

6. New features

New features will be prioritised and then implemented in order of priority. Features with UI consequences need to be grouped together as much as possible to ensure UI consistency and no-clutter.

- Enable pausing downloads for those browsers supporting it
- Enable direct-URL download for tools such as wget
- Authentication on the download URL

- Improved file upload handling 1: select and upload multiple files in one upload transaction (to one or more users, but same files go to all users!)
- Improved file upload handling 2: drag & drop
- Improved file upload handling 3: select and upload a folder in one upload transaction (a sort of auto-zip

feature)

- UI1: input validation checks on-input, for example email address
- UI2: display "characters left" counter for subject and message fields
- UI3: ensure users with visual impairment can use FileSender: ticket #349
- UI4: allow for different "first use" and "recurring use" views to allow "first use" to include for example an introduction video that you won't be bothered with lateron
- UI5: include "advanced features" functionality, hiding stuff that is not needed for simple operation of service
- UI6: allow for "two-click" use: a user ought to be able to send a file to himself with selecting a file followed by pressing send.

- Include addressbook functionality, possibly including groups, to make it easier to use FileSender for transporting files in longer-term collaboration relations

- Allow authorising of individual authorised users much in the same way we now authorise admins, preferably using through the admin web interface. Use case: allow only certain guest users coming from OpenIdP guest logons (user-whitelisting from open identity providers)
- Quota support, ticket #385

- Reliability: resumable uploads for uploading only missing chunks in case of large file upload error and to allow for pause/resume functionality (has GUI implications)
- Speed-up: parallelised upload-threads
- Support use from mobile platforms (iOS (iPhone, iPad), Android, Windows Phone, NOT Symbian) (belnet: low prio)
- Well defined and documented REST API, including AuthN/Auth needed to include FileSender in automated scientific workflow
- Integration with virus scanning (also needs database)
- Integration with time stamping service
- Config: automatic configuration file sanity checker

- Make FileSender entirely multi-tenant, both back-end and front-end, allowing for "cloudification" of a filesender install and service delivery to multiple organisations as well as per-institution branding if so desired.

7. Backend code improvements

- Code clean up

8. Make upgrading easy

- Ensure localised content (CSS templates, language files, config files) does not get overwritten on upgrade
- Devise good upgrade process requiring least manual steps possible

9. Set up automated testing and integrate it in the automated build process

- automated testing from a user's point of view against the automated builds
- simulate different popular browsers

- Linux (standardise on Ubuntu desktop + Scientific Linux?), Mac (version?), Windows7 (older versions?)
- automated testing of API functionality after API is implemented

10. Revise logging and statistics

- systematic analysis of what we can log and what we wish to log where
- improve statistics-oriented logging, making it easier to do per-institution logging

11. Project organisation

- devise version support lifecycle
- secure funding for 2012 development
- devise workplan 2012 (december 2011)