The FileSender project: integrated software dev. from PoC-code to package

Guido Aben\(^1\), Wendy Mason\(^2\), Paul Bonnington\(^2\) & Louis Moresi\(^2\)

\(^1\) AARNet Pty Ltd, \(^2\) Monash University

Intro slide #1: FileSender

…it’s the code under the hood of CloudStor
Intro slide #2: Factsheet

- Started April 2009
- 1.0 released 31 January 2011
- Core team of 6 people in 3 countries
- Agile and online collaboration

- Back: php+apache+postgresSQL+SimpleSAMLphp
- .deb, .rpm
- Front: Flash + HTML5 (for >2GB)
- Version 1.5: Flash-free HTML5 for >2GB uploads!

Talk overview

- How we used to go about SW dev
- What we’ve done differently this time
- Testing Regimes
- Workflows & Workflow Testing
- Measures of success: User Adoption
- Measures of success: Existing Installations
- Acknowledgments
How we used to go about SW dev

- Match hobby horse with internal need/lacuna
- Run survey; look for clues to back up hobby horse
- Propose thoroughbred horse to funding body
- Start in-house development straight away, using toolchain most convenient to developers
- Run resultant SW on server box of your taste
- Patch userunfriendliness with documentation
- Declare success*; move to new hobby horse, run service as-is, treat SW as abandonware

*at 100 users, at 200 logins, at 95% uptime, take your pick

What we’ve done differently this time

- Still started with a hobby horse of course!
- But then consulted with int’l eR / NREN orgs, exchanged ideas, inspected similar services and SW
- Reduced scope of initial planned release
- Turned our dev group outward, not inward
- Planned for ease of install, thus wide deployment, thus critical mass and sustainability
- Resisted urge to use comfortable/hip toolchain;
- Tested, retested and tested again: refused to fix with documentation; instead fixed user experience
Compare & Contrast

The start’s the same – a hobby horse.
It’s hard not to start with a hobby horse. You have to trust your instincts somehow

"It's really hard to design products by focus groups. A lot of times, people don’t know what they want until you show it to them.” – Steve Jobs

Old: Run survey;
look for clues to back up hobby horse

New: consulted with int’l eR / NREN orgs,
exchanged ideas, inspected similar services and SW
(even tried to get their code public-domained; weren’t successful, but would try again)

Our challenges are unlikely to be unique.
And our survey reach is limited.
Increasing the catchment area for “accidental discoveries” helped.
Surveying the same people again doesn’t, usually, for us.
If your challenge is really unique, perhaps you’re just scratching your own itch? Should that be funded with infrastructure monies?
Old: propose thoroughbred horse to funding body

New: Reduced scope of initial planned release

A tradition exists of hamming it up and then having to jettison ballast during the project. Might it be tickbox effect? Or one-upmanship?

Instead we chose to work on a pared-down initial release that we could pay for out of pocket. Lay the foundations first, don’t go overboard on featurism. If it’s great, we’ll contend for funding later.

Old: start in-house development straight away, using toolchain most convenient to developers Run resultant SW on server box of your taste (script, install binary dependencies, chroot and recompile at will)

New: turned our dev group outward, not inward Planned for ease of install everywhere, thus wide deployment, thus critical mass and sustainability Resisted urge to use comfortable/hip toolchain (decided against java; picked php, forbade ourselves to use external dependencies)

We made a conscious choice to sacrifice coding speed for long-term viability. The project was designed as an open source project; to have to be open about internal disagreement was... a learning curve. To have to refrain from grabbing a "more elegant" tool or language was hard at times. Having to wait for consensus... frustrating The result has been a much bigger userbase, more input, more corner case testing and a much reduced per-user cost
Old: Patch userunfriendliness with documentation
Then declare success*; move to new hobby horse, run service as-is, treat SW as abandonware
*at 100 users, at 200 logins, at 98% uptime, take your pick

New: tested, retested and tested again: refused to fix with documentation; instead fixed user experience

Testing Regimes

- **Nightly builds**

- **Workflow testing**
  - was manual up to 1.1 release
  - testing for 1.5 release partially automated using Selenium ([http://seleniumhq.org/](http://seleniumhq.org/)) (in progress)

- **Field testing**
Workflows - Overview


Workflows - User Steps

<table>
<thead>
<tr>
<th>AUTHENTICATED</th>
<th>GUEST</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. Upload</strong></td>
<td>Upload file using voucher</td>
</tr>
<tr>
<td><strong>2. Manage</strong></td>
<td>Resend email</td>
</tr>
<tr>
<td></td>
<td>Add new recipient(s)</td>
</tr>
<tr>
<td></td>
<td>Delete file manually</td>
</tr>
<tr>
<td></td>
<td>Delete unused voucher manually</td>
</tr>
<tr>
<td><strong>3. Receive</strong></td>
<td>Download files directly from “MyFiles” table</td>
</tr>
<tr>
<td></td>
<td>Download file</td>
</tr>
</tbody>
</table>

Key: Authentication required, optional step, Guest, Auto-generated email


**Workflows - Automatic Steps**

<table>
<thead>
<tr>
<th>FILES</th>
<th>VOUCHERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>File added to researcher’s “My Files” table</td>
<td>Voucher added to researcher’s Voucher table</td>
</tr>
<tr>
<td><em>url to uploaded file</em></td>
<td><em>url to voucher</em></td>
</tr>
<tr>
<td><em>file download notification</em></td>
<td><em>Voucher deleted automatically after single use</em></td>
</tr>
<tr>
<td><em>file deletion notification</em></td>
<td>manual deletion of unused voucher prior to expiry notification</td>
</tr>
<tr>
<td>File deleted automatically at expiry</td>
<td>Unused voucher deleted automatically at expiry</td>
</tr>
</tbody>
</table>

- Auto-generated email
- Automatic


---

**Workflow Testing - Access**

- support a range of popular OS/browser combinations
- installation only accessible via https (not http)
- detect required plugins, settings, feature support
- **no** access to full features for uploads using voucher or downloads
- auto-populated & required fields
- single, multiple & configured maximum email recipients separated by , or ;
- configured maximum expiry date ranges
- voucher deletion
Workflow Testing - Files

- use standard set of test files for replication
- ASCII files & UTF8 international character sets:
  - ISO-8859-1 e.g. blåbærsyltetøy (Norwegian)
  - non-ISO-8859-1 e.g. žćđš (Croatian)
- not accept selection of zero-sized or .exe files
- imposed/configured maximum file sizes
- cancel/pause & resume
- file deletion

Workflow Testing - Automation

- Selenium-IDE (Firefox browser plugin)
  - automation of file uploads, adding recipients & issuing vouchers (in progress)
  - other workflow aspects are still manual
- Selenium Server
  - AARNet hosted VMs are in place
  - Selenium-IDE scripts to be adapted to automate all workflow aspects
### Measure of Success: User Adoption

Statistics from AARNet’s FileSender installation:

- <1 helpdesk call per week (mostly re: AAF!)
- ~ 99.995% uptime
- ~4% of users sending >2Gb files, without instruction (seen 44Gb in the field)
- Since early Dec 2010, exceeding 1705 users & 4 TB uploaded

Stats are world-viewable at [https://cloudstor.aarnet.edu.au/mrtg/](https://cloudstor.aarnet.edu.au/mrtg/)

### Measure of Success: Existing Installations

- 14 known installations across 12 countries:
  - AARNet  Australia
  - ARNES  Slovenia
  - BELNET*  Belgium
  - FCCN  Portugal
  - HEAnet  Ireland
  - i2CAT  Catalonia
  - Internet2*  USA
  - Okinawa Institute of Science & Technology*  Japan
  - RESTENA*  Luxembourg
  - SIDN*  Netherlands
  - SRCE  Croatia
  - SURFnet*  Netherlands
  - TERENA*  Netherlands
  - UNINETT  Norway

* test installation

(see [https://www.assembla.com/wiki/show/file_sender/Existing_installations](https://www.assembla.com/wiki/show/file_sender/Existing_installations))
Acknowledgments

- **Current FileSender core development team:**
  - Jan Meijer, UNINETT - project leader
  - Guido Aben, AARNet - vice-project leader
  - Chris Richter, Ricoshae Pty Ltd - lead developer
  - Maarten Koopmans - developer
  - Xander Jansen, SURFnet - release manager & testing
  - Wendy Mason, Monash University - testing & documentation

- **Thanks to:**
  - UNINETT’s Feide RnD for guest memberships of Feide Open IdP
  - Alan Lo, VPAC for advice on selection & use of Selenium
  - Beta testers & users of FileSender installations for valuable feedback
  - Others who make contributions to FileSender code from time to time

- **More information:** [www.filesender.org](http://www.filesender.org)