Software Process Management

Project Report

MSc - Computer Science
Free University of Bolzano • A.A. 2010-2011

Group Members :

- Nicolò Paternoster [*9426*]
- Drón Khanna [*9420*]

RSS Owl

Yet another news reader

Nicolò Paternoster • paternoster.nicolo@gmail.com • Dron Khanna • dron_17@yahoo.co.in

Outline

Project Overview

What is RSSOwl?

Summary

- Project Name: RSS Owl
- Project Short Description: RSS Owl is an open source news reader (Atom, RSS)
- Current Version : 2.0.6
- Project Website : http://www.rssowl.org
- Platform : MacOs X, Linux and Windows
- License: Eclipse Public License
- Technologies : Build on Eclipse RCP 3.4.2 , db4o, Java

Short description

"Is an application that collect data from RSS-compliant sites are called RSS readers or aggregators. RSSOwl is such an application. RSSOwl lets you gather, organize, update, and store information from any compliant source in a convenient, easy to use interface, save selected information in various formats for offline viewing and sharing, and much more. It's easy to configure and the best of all: It's platform-independent." .

It is written in Java , and released under the Eclipse Public License , RSSOwl began as small project on Sourceforge at the end of July 2003. RSSOwl 1.0 was released on December 19, 2004. RSSOwl was the Sourceforge Project of the Month for January 2005 and by the end of 2009 the major release version 2.0 was announced and is now available to all users.

Our Contribution

We contribute to this open source project with **2 feature additions**, **3 feature modifications** and **1 feature deletion:**

Feature Addition

- *A1 :* Connect with the facebook account of the user and subscribe to facebook's friends news feed (Notification, Friends Status Updates, Links) .
- *A2 :* Read the emails from a *Gmail* account .

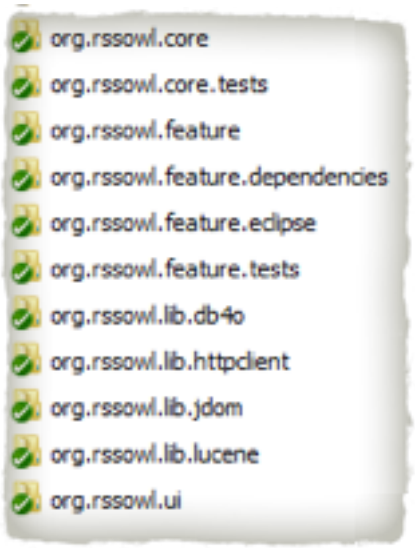
Feature Modification

- *M1 :* Give the user the possibility to directly listen to music into an rss item (*f.i.* podcast subscriptions with *enclosures)* using a flash-based open source player and adding it at the end of the item *description.*
- *M2 :* Add the Facebook "*like*" button to each item of a channel or to a channel itself.
- *M3 :* Modify the preference panel of RSSOwl putting into it some options related to our features.

Feature Deletion

- *D1 :* Delete the "*Mark as read*", "*Mark as Unread*" functions and menu entries.

Project Filesystem structure



User Stories

We *collected* different user stories for distribute the work among the **10 iterations** we had at disposal for developing the project.

List

ID	User Story Title	Description
Unique ID	Short and descriptive title for each user story	Description of the intent of the user story
1	Acquire Knowledge , Project exploring	The user can run the program under eclipse and see the application on facebook .
2	Authorization Panel creation (FB)	The user will be able to enter and save his Facebook credential . (A1)
3	Retrieve FB friends feeds	The user can view the feeds associated to Facebook updates. (A1)
4	Implement the gmail feed (pt. 1)	The user will be able to enter and save his gmail credential (A2)
5	Implement the gmail feed (pt. 2)	The user can read the feed channel associated with Gmail account.(A2)
6	Find an audio player	The user can try the audio player with a given URL outside RSSOwl.(M1)
7	Embed the player	If a mp3 file in enclosed into an rss item, then the user will be able to play it inside the product.(M1)
8	Add the FB "Like" button to items	The user will be able to use the Facebook "Like" button for each rss item. (M2)
9	Feature Deletion	The user cannot mark messages as "read" or "unread" .(D1)
10	Preferences pane	The user can edit some options about our features in the RSSOwl´s preference panel.(M3)

Priorities

As a good practice we gave to each user story a priority level, where 1 = must have, 2= ought to have, 3 = nice to have . We balanced the different level of priorities in order to have 4 US of priority 1, 3 US of priority 2 and 3 US of priority 3 as in the following table.

US ID	Priority
1	1
2	1
3	1
4	3

US ID	Priority
5	3
6	2
7	2
8	3
9	1
10	2

Effort estimation

Remarkable informations: An overview of the time estimation of the project

Total Actual time : **106.5 hrs**

Total Time Estimated : 107.5 hrs

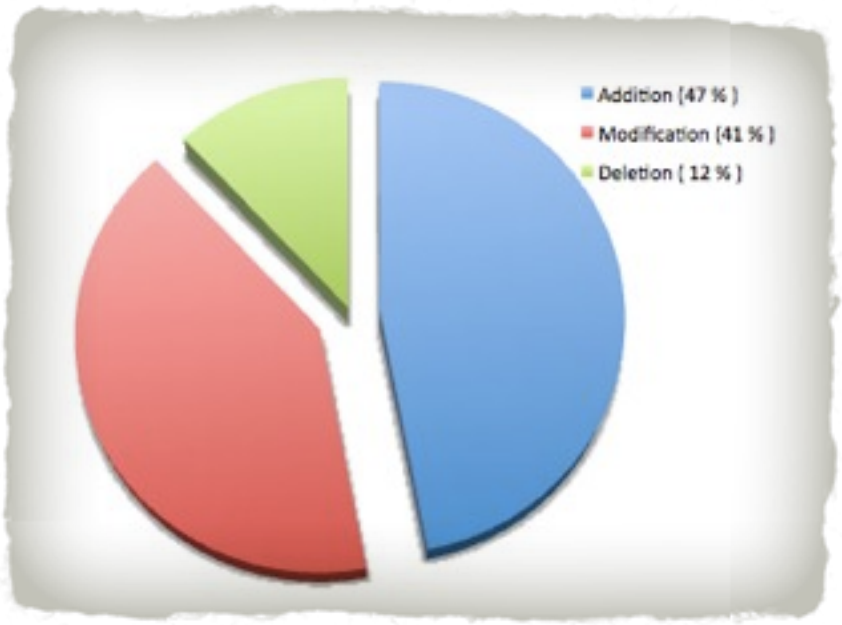
Total gap (absolute value) : 10 hrs

Average gap per iteration: 1 h

Effort by feature:

ID	Feature description	Actual time (hrs)	Total Percentage
A1	Facebook Feeds	34	36.96%
A2	Gmail Feeds	8.5	9.24%
M1	Audio player	20	21.74%
M2	Facebook like button	9	9.78%
M3	Preference Panel	10.5	11.41%
D1	Mark as read/unread	10	10.87%

Effort by addition , modification, deletion:



Code Revisions

An overview of the repository usage

File types

Type	Files	LOC	LOC per file
*.java	698 (23.9%)	172624 (61.6%)	247.3
*.xml	257 (8.8%)	77495 (27.7%)	301.5
*.xpm	1 (0.0%)	5747 (2.1%)	5747.0
*.exsd	22 (0.8%)	2721 (1.0%)	123.6
*.properties	48 (1.6%)	2707 (1.0%)	56.3
*.prefs	26 (0.9%)	1712 (0.6%)	65.8
*.html	21 (0.7%)	969 (0.3%)	46.1
*.txt	7 (0.2%)	829 (0.3%)	118.4
*.opml	12 (0.4%)	827 (0.3%)	68.9
*.nsi	1 (0.0%)	354 (0.1%)	354.0
Others	46 (1.6%)	14133 (5.0%)	307.2
Non-Code Files	1781 (61.0%)	0 (0.0%)	0.0
Totals	2920 (100.0%)	280118 (100.0%)	95.9

Repository

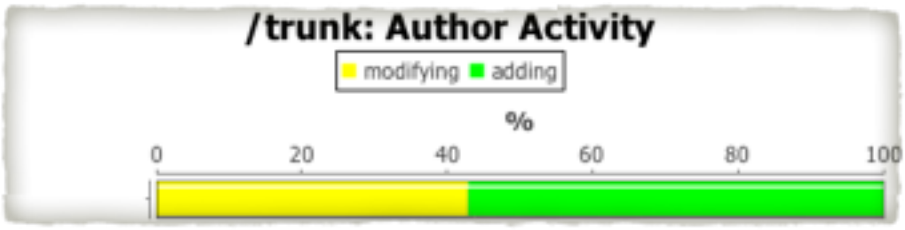
Total revisions: **160**

Average number of revision per sprint: **16**

Most revised files

File	Revisions
org.rssowl.ui/src/org/rssowl/ui/internal/actions/FacebookAuthenticationAction.java	21
org.rssowl.ui/plugin.xml	14
org.rssowl.core/src/org/rssowl/core/connection/FacebookAuthentication.java	12
org.rssowl.ui/plugin.properties	11
org.rssowl.ui/src/org/rssowl/ui/internal/dialogs/messages.properties	11
org.rssowl.ui/src/org/rssowl/ui/internal/actions/FacebookProgressListener.java	11
org.rssowl.ui/src/org/rssowl/ui/internal/dialogs/FacebookCredentialDialog.java	10
org.rssowl.core/src/org/rssowl/core/util/SyncUtils.java	10
org.rssowl.core.tests/src/org/rssowl/core/tests/UITests.java	10
org.rssowl.core/src/org/rssowl/core/connection/Messages.java	9
org.rssowl.core/src/org/rssowl/core/connection/messages.properties	9
org.rssowl.core/src/org/rssowl/core/internal/interpreter/AtomInterpreter.java	8
org.rssowl.ui/src/org/rssowl/ui/internal/dialogs/Messages.java	8
org.rssowl.core.tests/src/org/rssowl/core/tests/LocalTests.java	8
org.rssowl.ui/src/org/rssowl/ui/internal/handler/facebookButtonHandler.java	8
org.rssowl.core/src/org/rssowl/core/persist/INews.java	7
org.rssowl.core/src/org/rssowl/core/internal/interpreter/RSSInterpreter.java	7
org.rssowl.ui/src/org/rssowl/ui/internal/Activator.java	7
org.rssowl.core.tests/src/org/rssowl/core/tests/controller/ReloadTestLocal.java	6
org.rssowl.core/src/org/rssowl/core/internal/persist/News.java	6

Lines of Code overview



Package name	Original (LOCs)	Before Deletion (LOCs)	Final (LOCs)
core	49254	49821	49698
UI	82372	83543	82874
Test	96058	96991	95932
total	227684	230355	228504

Total Added LOCs	Total Deleted LOCs
2671	1851

Classes

Added	Modified	Deleted
19	28	9

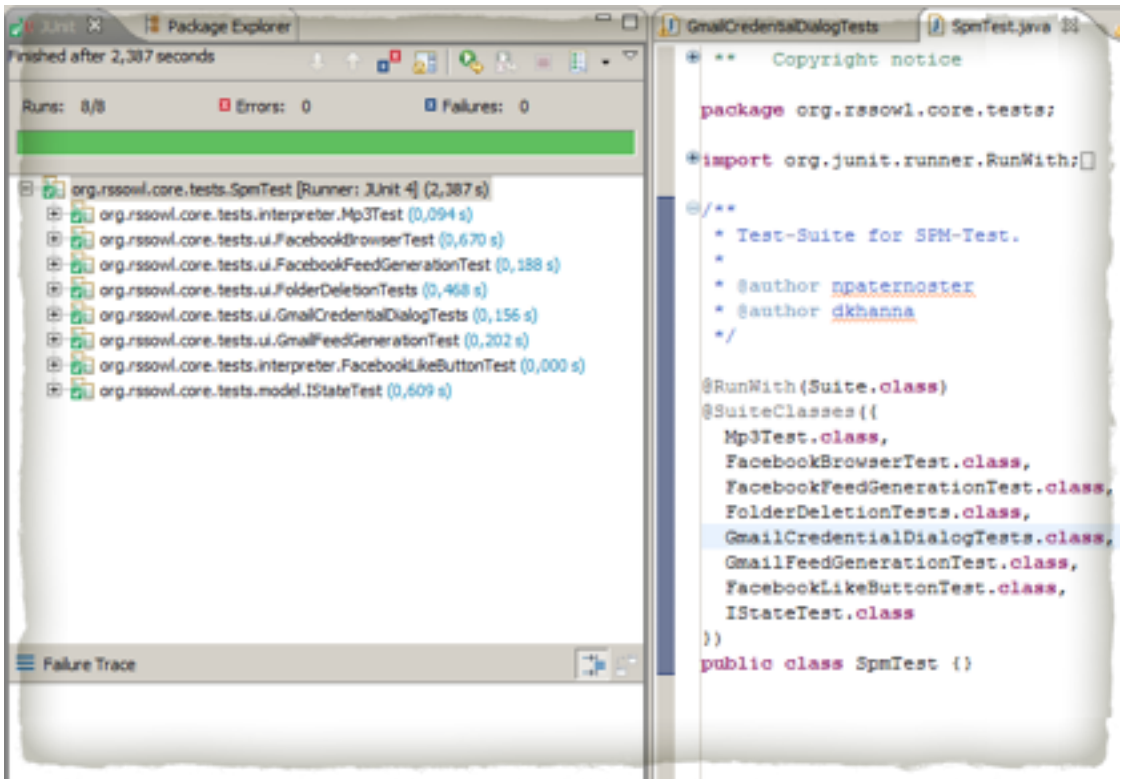
Testing

Statistic from JUnit

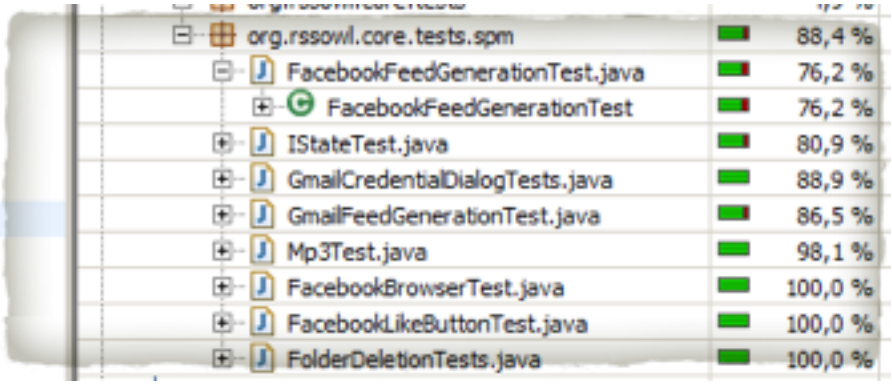
Existing tests

The original project had already a big set of unit test well developed, located under the package *org.rssowl.core.tests* .

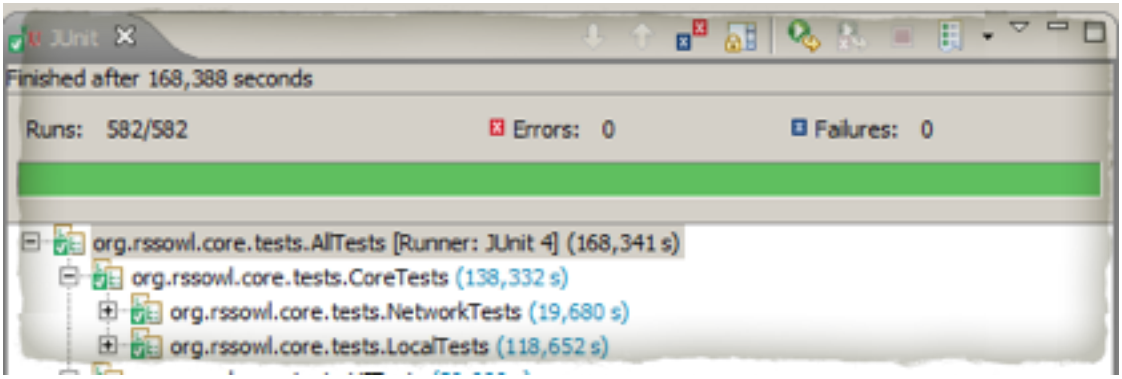
We created a sub-package containing our tests: *org.rssowl.core.tests.spm*



With this test we achieved an high percentage of code coverage : **88,4 %**



All other existing tests are passing after our integration.



Tools

Which tools we used in this project

- Eclipse 3.4.2 for RCP - http://www.eclipse.org/downloads/packages/eclipse-classic-342/ganymedesr2
  - Delta Pack - http://www.eclipse.org/downloads/
  - Subclipse - http://subclipse.tigris.org/
  - EclEmma - http://www.eclemma.org/
- Argo UML - http://argouml.tigris.org/
- Tortoise SVN - http://tortoisesvn.tigris.org/

- Rapid SVN - http://rapidsvn.tigris.org/
- StatSVN - http://www.statsvn.org/
- WinMerge - http://winmerge.org/
- loc-calculator - http://code.google.com/p/loc-calculator/
- Open Source Flash Mp3 Player - http://flash-mp3-player.net/
- Facebook Like Button - http://developers.facebook.com/docs/reference/plugins/like

Acceptance tests

All the acceptance test we wrote can be performed correctly.

- The user can browse the facebook page associated with our application after we provided the link. (U.S. ID : 1)
- Clicking the *Facebook* button on the RSSOwl's toolbar a Dialog will show up, and the user can start the authorization procedure clicking the "OK" button. (U.S. ID : 2)
- After clicking the "OK" button in the Facebook Dialog, the user will be asked to insert *fb username* and *password* and allow the RSSOwl fb application to access his information. (U.S. ID : 2)
- If the user gives to the facebook application permissions, clicking the "Allow" button, the procedure to get the users feed will automatically start and after a few seconds he can see on the side bar a new folder called as *Facebook feeds* with feeds connected to his account. (U.S. ID : 3)
- If the user has already gave fb permission and got the feeds ,when he click again the *Facebook* button, nothing will happens.(U.S. ID : 2)
- If the user has given proper *facebook* authentication after clicking the *save* button he can see on the side bar a new folder called as *Facebook feeds* with feeds connected to his account. (U.S. ID : 3)
- Clicking the *Gmail* button on the RSSOwl's toolbar the user will be asked to insert *username* and *password* for their gmail credential in the forms and save it using the *save* button . (U.S. ID : 4)

- If the user has already saved the data ,when he click again the *Gmail* button he will see the username and the masked password that he entered before. (U.S. ID : 4)
- If the user has given proper *gmail* authentication after clicking the *save* button he can see on the side bar a new folder called as *Gmail* with feed connected to his gmail account. (U.S. ID : 5)
- Providing a valid URL that points out to an mp3 file, the user will be able to play that file within an HTML page (given by us) with the open source player .(U.S. ID : 6)
- When a rss item has a *rss enclosure* with the *mp3* extension, if the client is connected to the Internet,the user can play the song within RSSOwl (U.S. ID : 7)
- When the user has already entered his valid facebook credential,after clicking the *facebook like button* integrated in the item, he will see the *activity* related to this action on his *facebook wall. (U.S. ID : 8)*
- The user will not be able to mark a message as *read* or *unread*. (U.D. ID : 9)
- The user can edit some preferences about new implemented features and changes will affects new items (U.D. ID : 10)

Final Release

A multi-platform application

Our final product is a multi-platform stand alone application .

File sizes

Windows Platform	MacOsX	Linux gtk
21.7 Mb	22.4 Mb	22.1 Mb