Contributing to Eclipse: A Case Study

Henttonen Katja, Matinlassi Mari
VTT Technical Research Centre of Finland
P.O. Box 1100, 90571 Oulu, Finland
{Katja.Henttonen, Mari.Matinlassi}@vtt.fi

Abstract: Open source software has gained a lot of well-deserved attention during the last few years. Eclipse is one of the most successful open source communities providing an open development environment and an application lifecycle platform. The main aim of this paper is to describe a case study on contributing to the Eclipse open source community and report experiences. The most important experiences are related to building an architecture model repository tool as an Eclipse plug-in and starting a new community around it.

1 Introduction

Open source software (OSS) has been growing its popularity among software developers, communities and media over a decade and lately also the research community has been active in studying the subject. Among the most successful open source projects, Eclipse (www.eclipse.org) is an open development platform and application framework for building software. The Eclipse community has distinguished itself in productivity and creativity [Ki05] and has developed new features that have evolved Eclipse towards a platform that is integrating not only tools but also applications and services [Gr05]. The new features also make Eclipse a strong force in the embedded market [Er06] [Kl05] and make the community even more international [La05].

There are several ways of contributing to open source communities, e.g. fixing bugs and providing new features. In this paper, contributing to Eclipse is considered as a twofold issue. On the one hand, a contribution, i.e. a plug-in, needs to be developed, and on the other hand, it needs to get published to the audience: users and community [Be04]. A good plug-in is modular, because the only way an open source project can mature is by allowing a number of people to participate in the development [Fl01]. While developing modular plug-ins is relatively straightforward [Ga04], the acute question remains how to get the modular plug-in to the open source "market"? The contribution of this paper is to introduce a case study on contributing to Eclipse and to report the experiences on the subject. The rest of the paper is structured as follows. The next section summarizes the approach used for contributing to Eclipse. After that the case study is introduced and the experiences discussed. Conclusions close the paper.

2 Approach for contributing to Eclipse

Open source projects have been likened to a bazaar [Ra03]. An open source project gathers people, whose skills, motivation and time of involvement may vary significantly [GaA04], to work in a distributed environment. Any contributor is welcomed to add a new feature to "scratch an itch". In such a development model, the importance of maintainability and the ease of making modifications are highlighted. This goal is achieved, e.g., by designing a modular software architecture.

In order to create a successful open source community around an Eclipse plug-in, the contributor needs (1) to build a plug-in with well designed software architecture, (2) to enable others to extend the plug-in with further plug-ins, and (3) to publish the plug-in as an open source project. The main stakeholders of the Eclipse community [Be04] are summarized in Table 1. In our approach for contributing to Eclipse, and in this case study, we are acting in three different stakeholder roles: extender, publisher and enabler. In this hierarchical stakeholder stack, the stack is built from a technical point of view. That is, publishing an Eclipse extension does not require making it extendable. Although this is true, attracting an active open source community, as stated above, requires easily expandable architecture, which in this case is implemented by Eclipse extension mechanisms.

Table 1. Stakeholders in the Eclipse community.

Stakeholder	**Description**
User	Uses Eclipse as it is.
Configurer	A user who customizes his/her experience of Eclipse within the limits envisioned by the original programmer.
Extender	A programmer who makes extensions by plugging in functionality.
Publisher	An extender who makes extensions available to others for loading them.
Enabler	A publisher who has defined one or more extension points for a plug-in, thus enabling others to extend the contribution.
Committer	Modifies the Eclipse code and incorporates changes into the global Eclipse release. Requires trust of existing committer community.

An open source project can be announced when the code artifact has been implemented into a "minimal but working version" [Go05]. Mandatory requirements, which need to be implemented before announcing the project, were identified in the requirement specification phase. In addition to the code artifact, an open source project needs a standard set of tools for managing information, including at least a website, mailing lists, a code versioning system and a real-time chat room [Fo05]. Once the infrastructure is ready, the next goal is 1) to get publicity for the project, 2) to gain a large enough user community, and 3) to gradually increase the level of community involvement. To be successful, an open source project requires marketing just like any other product. [Go05]

3 Case study: Stylebase for Eclipse

3.1. Overview

Quality-driven architecture design is an approach for software architecture design which emphasizes the importance of quality attributes (e.g. performance, modularity or maintainability) during the development[1]. The approach relies on the assumption that architectural styles and patterns, and also design patterns, embody different quality attributes. When patterns are applied in the architecture, the quality characteristics of the selected patterns are reflected in the entire software architecture. Stylebase is an important part of the quality-driven software architecture design and analysis (QADA®) methodology. Stylebase is a knowledge base of architectural styles and patterns [Ni05], and design patterns. The aim of the stylebase is to assist the architect in selecting styles and patterns which promote desired quality attributes. On the other hand, the stylebase may also assist in evaluating software architectures [Do02].

The Stylebase for Eclipse community has developed an open source implementation of the stylebase based on previous work [Me05] [MN05], and it has also published as a tool for browsing and maintaining the stylebase. The tool can be used in three ways: 1) as an electronic library for patterns, 2) as a guide for evaluating quality-driven architecture and 3) as a guide for quality-driven architecture modeling. When used as an electronic library, the architect browses the stylebase as a pattern catalogue. When used for model evaluation, the architect detects which patterns have been used in an architecture model and then checks from the stylebase which quality-attributes are associated with these patterns. When constructing a new architecture model, the architect searches the knowledge base and selects patterns according to the desired quality characteristics.

3.2. Expandable plug-in architecture

In the Stylebase for Eclipse, modularity was implemented at two levels. (1) The internal architecture of the Stylebase for Eclipse was designed to be modular. This was achieved by following the well-known model-view-controller pattern (MVC), see e.g. [Bu96]. (2) The Stylebase for Eclipse provides functionality which lets users build custom extensions without touching the source code of the Stylebase for Eclipse. This has been achieved by (a) building *access points* by implementing and exporting API (Application Programming Interface) packages, and (b) defining *extension points* with the Eclipse extension point mechanism. The access and extension points are listed in Table 2 and Table 3 respectively.

[1] http://virtual.vtt.fi/qada/

The MVC architecture (Model, View and Controller) is a way of breaking an application into three parts: model, view and controller. The user input, the manipulation of data and the visual feedback to the user are separated and handled by controller, model and view objects respectively. The MVC architecture was selected for the following reasons:

1) MVC promotes extensibility by allowing multiple representations (views) of the same information (model). This makes it easy to update the graphical user interface, which is especially prone to change requests, and/or to customize views based on user profiles.

2) MVC promotes code reuse by allowing a single view to show data from different models. By adding a new model, it is possible to adjust the tool to manage entirely different types of data with minimum recoding effort.

3) MVC facilitates maintenance by allowing an individual developer to focus on one aspect of the application at a time. Multiple developers can simultaneously update the interface, logic or input of an application without affecting other parts of the source code.

5) MVC architecture suits well for Eclipse plug-in development because the view classes of Eclipse can receive any other class as an input object. The Eclipse platform itself has a model-view-controller architecture [Gr04].

In order to increase the level of modularity, the three main components communicate with each other via predefined interfaces. Figure 1 shows how the plug-in implements a model-view-controller pattern.

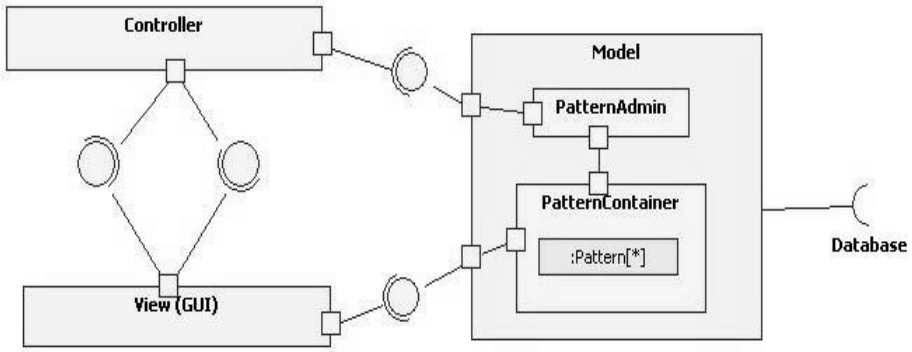

FIGURE 1: Stylebase plug-in composite structure

Tables 2 and 3 describe functionality that the Stylebase for Eclipse provides for downstream plug-ins.

Table 2. The access points provided by the Stylebase for Eclipse

Controller	The interface provides access to the control component. It lets users associate the functions of Stylebase for Eclipse with the GUI of another plug-in.
Model	The interface gives access to the model component. It provides a set of methods for retrieving and updating essential data in the Stylebase.
Database (SQL)	The interface provides SQL-level access to the underlying database. It helps in implementing specific functionality not provided by the model interface.

Table 3. The extension points provided by the Stylebase for Eclipse

Model Extension Point	The extension point provides the means of adding new models, units for storing and handling different types of data.
GUI Extension Point	The extension point provides the means of customizing the user interface of the Stylebase for Eclipse. It allows users to add their own views and/or menu items to the main view of the Stylebase for Eclipse. The controller component is extended respectively.

3.3. Founding a community

In the case of Stylebase for Eclipse, , the founding of the community was started right after implementing the mandatory functionality. Considering our limited resources and the laborious effort of self-hosting a project, it was decided to subscribe to one of the websites providing free hosting services for open source projects. Such services include web hosting, mailing lists, code archive, file hosting and bug tracking. For selecting hosting facilities, the most well-known, general-purpose hosting facilities – SourceForge.net, Savannah, BerliOS and Google code – were evaluated. They all run a collaborative software development management system originally called "SourceForge". SourceForge.net uses a closed-source proprietary version of this software, while the three other websites run free versions forked from the last open source version of SourceForge.

When registering a project to any such facility, the submitter has to state which license the project uses. We decided to use the GNU (**G**NU's **N**ot **U**nix) General Public License (GPL), which has been written by the Free Software Foundation. Many contributors are familiar with it as it is the most widely used open source license [Fo06]. GPL efficiently prevents creation of closed-source forks [Go05] and ensures that our software can be unambiguously used with MySQL client libraries which are also GPL licensed. The main drawback of GPL is that it may turn companies away from contributing to a project [Go05]. Another disadvantage is the incompatibility with EPL (Eclipse Public License) [Fr06] which, however, may be solved by GPL version 3 [Or06].

Savannah provided by the Free Software Foundation was selected as a project hosting provider because it is ad-free and it provides the most professional looking user-interface. Unlike the other alternatives, Savannah is dedicated to free software advocacy and has strict hosting policies, such as code review before accepting a new project as a Savannah project. The Savannah administration requires that all hosted software 1) is licensed under a free software license compatible with the GNU General Public License, 2) runs on at least one free operating system such as Linux, and 3) is not dependent on any non-free software. Non-free software includes all proprietary programs (such as Macromedia Flash and the Microsoft SQL Server), non-free file formats (such as the Graphic Interchange Format), and non-free programming languages (such as the Sun implementation of Java). [FrA06]

In order to express support to the Eclipse community, it was decided to apply for the membership of the Eclipse Foundation. Associate membership is free of charge for non-commercial entities like universities and research institutes. The Eclipse Foundation requires that associate members commit to 1) deliver added value to the Eclipse community within 12 months and 2) publicly announce joining the Eclipse Foundation within 90 days [EcA06]. While the associate membership does not grant decisive power, it grants rights to participate in project reviews, project creation and discussions on Eclipse intellectual property policy. It also entitles the member to use the Eclipse Foundation Member logo in marketing activities. [Ec06]

In order to make the project known to the public, we decided to publish announcements on some popular, open source related websites and to send one-time postings to carefully selected mailing lists. We chose not to send email to several, large mailing lists because such postings are easily considered as spam. As well as regular internet users, open source developers dislike irrelevant information that is blocking their communication channels [Ra06]. Further, we promoted the project at various open source related events, such as the OpenMind seminar (http://www.openmind.fi/). A brochure and poster were designed for marketing the project. We also contacted lecturers of software development courses at the University of Oulu and offered the Stylebase for Eclipse project as a practical work for their students.

The Eclipse Plug-in Central (EPIC) is obviously a website where we wanted our plug-in to be listed. This is a place where Eclipse users search for commercial and open source plug-ins. An announcement was also placed at a few open source development portals, at FreshMeat.net, for example, which Fogel [Fo05] mentions as the number one place to be seen.

5 Experiences

5.1. Starting up the project - project hosting

As stated above (see Section 3.3), we evaluated four project hosting services and selected Savannah as the most appropriate one for our purposes. Savannah administration had very strict requirements on how the GNU General Public License should be applied to the code artifact. The requirements include placing copyright and free software statements both at the beginning of each source file and in a README file in every subdirectory containing binary files. According to the Free Software Foundation, this is the only way to ensure that the code is effectively protected by the license [FrB06]. What made things inconvenient was extending the requirement to apply to all external libraries distributed at Savannah. In the case of the Stylebase, this meant, for example, that one needed to open the JAR (**J**ava **Ar**chive) file of the MySQL client, place a README file into dozens of sub directories and then to package the source code and binaries back into a JAR file. Savannah also required that the source code of all external libraries was distributed with the plug-in. It was not sufficient to provide instructions on where to download the source code [RoA06].

Some interesting lessons were learned with different implementations of Java class libraries. Until November 2006, Sun Java libraries were published under Sun Community Source License (SCSL), which is a hybrid between an open source license and a traditional proprietary license [Ga6]. Many free software developers were unhappy with the Sun licensing scheme [So05] and Free Software Foundation coordinated the development of GNU Class Path, a GPL licensed implementation of Java. As the Stylebase plug-in was developed on Sun platform, some Sun-only features were inadvertently used. Later on, it was required to redo these features because the code could not be compiled with GCJ (GNU Compiler for Java), a widely-used open source Java compiler. In November 2006, as a response to many requests, Sun finally decided to publish its Java implementation under GNU General Public License [Su06]. Time shall show whether the decision will diminish or increase differences between Java implementations. In any case, making an open source product dependent of one particular implementation would hardly be a good choice.

The application process to Savannah took longer than planned. This was most probably due to the following reasons. Not all the requirements for application were clearly stated at once, but rather mentioned one after another each week. The voluntary nature [Go05] of open source communities was really put into action when our emails were replied to mostly on weekends and there was often a long delay before receiving a reply. Due to the strict deadline of publishing our open source project we had to set up a site at SourceForge.net where the application was processed and accepted in less than a day. Considering the amount of bogus projects found at SourceForge.net, the quick approval appears to reflect a loose hosting policy rather than efficiency.

5.2. Integrating OS components - license issues

At the time the Stylebase for Eclipse was about to be published, the Stylebase for Eclipse tool was dependent on an XML (eXtensible Markup Language) object model called JDOM, which was published under its own open source license specifically written for the JDOM project. Due to the fact that JDOM uses a permissive license similar to the MIT (Massachusetts Institute of Technology) license, we considered it to be GPL-compatible. However, Savannah refused to accept the JDOM component as a part of the project. The clause that prohibits the use of the name JDOM in derivative works can be seen as an additional restriction and this would rule out the GPL-compatibility [RoB06]. A polite posting [He06] to an open source forum regarding the GPL-compatibility of JDOM caused an intense "flame war". It seems that whenever a project uses its own license, compatibility issues are a matter of debate. The JDOM component was therefore replaced with another document object model which had been published under the GNU Lesser General Public License (LGPL). The Open Source Watch [Os06] recommends that GPL-licensed software be combined with licenses that are listed as GPL-compatible on the website of the Free Software Foundation.

5.3. Attracting users and contributors - marketing activities

At the time of writing this paper, not all the marketing activities mentioned above (Section 3.3.) were executed. However, initial experiences were collected as follows. Applying for the membership of Eclipse Foundation was quite straightforward. However, while reviewing the membership agreement of Eclipse Foundation, it was unclear whether contributions would be accepted under any other open source license than the Eclipse Public License (EPL). The Eclipse Foundation confirmed that the strict EPL requirement applies only to contributions which are part of the official Eclipse project and hosted at eclipse.org [Sm06]. Soon after that, the application for Eclipse Foundation Associate Membership was accepted [Mc06].

The most effective marketing activity seemed to be giving a demonstration at a seminar targeted for special audience interested in open source. A concrete metric set for assessing the effectiveness of marketing activities was provided by the number of downloads at stylebase.sourceforge.net. A few days after the demonstration we were able to witness nearly 50 downloads from different IP addresses.

6 Conclusions

The aim of this paper was to describe a case study on contributing to the Eclipse open source community and to report some experiences. The initial experiences were reported soon after the official publication of the project in autumn 2006. The case study was concerned with a software architecture tool called Stylebase for Eclipse, which was licensed under the GNU GPL and implemented as an Eclipse plug-in. Developing Eclipse plug-ins was quite straightforward and therefore the most interesting experiences were related to (1) licensing issues while integrating several open source components together, (2) project hosting services while starting up an open source community, and (3) marketing activities while attracting users and contributors for the community.

Based on the case study, the most important conclusions are as follows. First, Savannah by the Free Software Foundation was selected as a project hosting provider because it was ad-free and provided the most professional looking user-interface. One of the experienced advantages of Savannah was that it would host only carefully reviewed projects (causing a disadvantage of long acceptance process). We also tried out another option - SourceForge – where, although the acceptance process was very fast, the responsibility of project quality was left for the respective community.

Second, it seems that whenever an open source subcomponent uses its own license written especially for the project, compatibility issues will be a matter of debate. Resolving compatibility issues takes time and therefore should be taken into account in schedules, or in our case regarding a GPL license, one should only use open source components with GPL compatible licenses accepted by the Free Software Foundation.

Third, the marketing of an open source project was deemed to be crucial in order to gain users and contributors for your project. We used several marketing channels such as passing information down by word of mouth, handing out brochures, and demonstrating the proposed tool at a seminar for open source related audience. Marketing activities caused an immediate tenfold increase in code downloads at the project website.

Acknowledgements

The publication of this paper has been supported by Eureka ITEA research project COSI funded by the National Technology Agency (Tekes) and VTT.

References

[Be04] Beck, K.; Gamma, E.: Contributing to Eclipse. Dr. Dobb's Journal. 29, 9, 2004; P. 74-8.

[Bu96] Buschmann F; Meunier R; Rohnert H; Sommerlad P; Stal M: Pattern-oriented software architecture - a system of patterns. Wiley, 1996.

[Do02] Dobrica, L.; Niemela, E: A survey on software architecture analysis methods. Software Engineering, IEEE Transactions on 28(7): 638-653. 2002.

[Ec06] Eclipse Foundation: Eclipse Rights by Membership Category http://www.eclipse.org/membership/become_a_member/How2Join%20Eclipse%20Rights%20by%20Membership%20Category.pdf. Visited 26.10.2006.

[EcA06] Eclipse Foundation: Eclipse Membership Application or Change in Representation. http://www.eclipse.org/membership/become_a_member/Membership %20Application.pdf. Visited 26.10.2006.

[Er06] Erickson, J: Eclipse Foundation Releases Embedded RCP. The World of Open Source. Available at http://www.ddj.com/blog/opensourceblog/archives/2006/10/eclipse_release.html

[Fl04] Fleury, M.; Lindfors J.: Enabling component architectures in JVMX. http://www.onjava.com/pub/a/onjava/2001/02/01/jmx.html. 2001. Visited 30.6.2006.

[Fo05] Fogel, K.: Producing Open Source Software. How to run a successful free software project. O'Reilly, 2005; P. 34, 43, 45-47.

[Fr06] Free Software Foundation: GPL-incompatible Free Software Licences. http://www.fsf.org/licensing/licenses/index_html#GPLIncompatibleLicenses. Visited 20.10.2006.

[FrA06] Free Software Foundation: Savannah Services and Requirements https://savannah.nongnu.org/register/requirements.php Visited 26.10.2006

[FrB06] Free Software Foundation: Frequently Asked Questions on GNU GPL: Using the GPL for your programs. http://www.gnu.org/licenses/gpl-faq.html. Visited 27.10.2006.

[Ga04] Gamma, E.; Beck, K.: Contributing to Eclipse - Principles, Patterns, and Plug-Ins. Addison Wesley, 2004; P. 395.

[GaA04] Gacek, C.; Arief, B.: The many meanings of open source. IEEE Software. 21, 1, 2004; P. 34-40.

[Ga06] Gabriel,R; Joy, W: Sun Community Source Licensing (SCSL) – Principles. http://www.sun.com/software/communitysource/principles.xml. Visited 26.10.2006.

[Go05] Goldman R.; Gapriel R: Innovation Happens Elsewhere. Open Source as Business Strategy. Elsivier, 2005; P. 15-16, 157, 190-191, 224, 256.

[Gr04] Griffin, C: Transformations in Eclipse. The proceedings of 18th European Conference on Object-Oriented Programming. Norway, 2004.

[Gr05] Gruber, O. et al.: The Eclipse 3.0 platform: adopting OSGi technology. IBM Systems Journal. 44, 2, 2005; P. 289-99.

[He06] Henttonen, K.: Incompatibility with GPL [email]. Message to: jdom- interest@jdom.org. 28.9.2006. Cited 26.10.2006. Available at http://www.jdom.org/pipermail/jdom-interest/2006-September/015549.html

[Ki05] Kidane, Y.; Gloor, P: Correlating Temporal Communication Patterns of the Eclipse Open Source Community with Performance and Creativity. NAACSOS Conference, June 26 - 28, Notre Dame IN, North American Association for Computational Social and Organizational Science, 2005. Available at http://www.ickn.org/documents/Naacsos_Kidane_Gloor.pdf

[La05] Lammers, D.: Tool developers rally around Eclipse. Electronic Engineering Times. 1369, 2005; P. 47-49.

[Mc06] McGaughey, S.: Welcome VTT Technical Research Center of Finland to Eclipse Foundation as an Associate Member [email]. Message to Katja Henttonen. 4.10.2006. Cited 1.11.2006.

[Me05] Merilinna, J: A Tool for Quality-Driven Architecture Model Transformation. Espoo, VTT. VTT Publications. 2005. http://www.vtt.fi/inf/pdf/publications/2005/P561.pdf

[MN05] Merilinna, J; Niemelä, E: A stylebase as a tool for modelling of quality- driven software architecture. Proceedings of the Estonian Academy of Sciences. Engineering December 2005. Special issue on Programming Languages and Software Tools. vol. 11, 4. 2005

[Ni05] Niemelä, E.; Kalaoja J. : Toward an architectural knowledge base for wireless service engineering. IEEE Transactions on Software Engineering 31(5): 361 - 379. 2005.

[Os06] Open Source Watch: What is open source software? http://www.oss-watch.ac.uk/resources/opensourcesoftware.xml. Visited 27.10.2006.

[Or06] O'Riordan, C: The Transcript of the Speech by Eben Moglen (Section 7e) at the Opening Session of the First International GPLv3 Conference on January 16th 2006. Available at http://www.ifso.ie/documents/gplv3-launch-2006-01-16.html.

[Ra03] Raymond, E: The Cathedral and the Bazaar. O'Reilly, 2001. Available at http://catb.org/~esr/writings/cathedral-bazaar/.

[Ra06] Raymond, E.; Moen R: How to Ask Smart Questions The Smart Way. http://catb.org/esr/faqs/smart-questions.html. Visited 27.10.2006.

[Ro06] Robson, S.: [task #5865] Submission of Stylebase [email]. Message to Katja Henttonen and savannah-register-public@gnu.org. 24.9.2006. Cited 26.10.2006. Available at http://www.mail-archive.com/savannah-register-public@gnu.org/msg06275.html

[RoA06] Robson, S.: [task #5865] Submission of Stylebase [email]. Message to Katja Henttonen and savannah-register-public@gnu.org. 12.10.2006. Cited 26.10.2006. Available at http://www.mail-archive.com/savannah-register-public@gnu.org/msg06347.html

[RoB06] Rowan, W.: Open Source Development - An Introduction to Ownership and Licensing Issues. University of Oxford, 2006. Available at http://www.oss-watch.ac.uk/resources/iprguide.xml.

[Sm06] Smith, D.: RE: Question on Eclipse membership [email]. Message to: Katja Henttonen. 2.10.2006. Cited 26.10.2006.

[So05] Souza, B.: How Much Freedom Do you Want? In compilation Cris Dibona & al. (edit): Open Sources 2.0. O'Reilly, 2005; P219-224.

[Su06] Sun MicroSystems Inc. Sun Open Sources Java Platform and Releases Source Code Under GPL License Via NetBeans and Java.net Communities. http://www.sun.com/smi/Press/sunflash/2006-11/sunflash.20061113.1.xml. Visited 28.12.2006.