Integrated Open-Source Software Development Activities Browser CoxR

Makoto Matsushita, Kei Sasaki, Yasutaka Tahara, Takeshi Ishikawa and Katsuro Inoue
Graduate School of Information Science and Technology, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan

Abstract

Open-source software and its style of software development activities [1] are getting attentions to the world[5]. Usually open-source software development itself is also open, using network-wide tools such as software repositories, email archive and associated search engine, web system, etc. We can check development activities using such tools, however, each tools are independent each other so it is very hard to understand the relationships between activities. In this paper, we propose browsing system **CoxR***, for open-source software development activities, and show the* **CoxR** *prototype. This system consists of two parts, the one supports to understand relationships between source code changes and email discussion, and the another one supports to search source code changes. With this browser, open-source developers can grasp the whole activities of software development project.*

1. Open-Source Software Development

In the last decade of 20th century, a brand-new style of software, called "open-source software" was getting attentions from the world. In most cases, open-source software is a software which code are free-to-access to the public, and lots of people in the world may join to activities of development such as writing a new code, adding a new features, fixing a bug, proposing a new architecture, discussing a current design problem, and so on.

There are some tools used in open-source software development (OSD for short) environment. CVS is a product repository system and manages revision history of each file by check-in/check-out model [2]. Email is a pretty-common tool for communicating between developers. Since a variety of development environment and skill of each developer, other development tools such as CASE tool, video conferencing tools are hard to employ to this type of development. As a result, CVS repository and email archive are the core storage for OSD.

1.1. Typical Development Styles

It can be considered that whole OSD is a traditional spiral-model with very short cycle, and each cycle are driven by independent developers' parallel activities which are a series of small task. Each task is roughly categorized into two area, "writing a product by oneself" and "communicating with other developers." just like parallel-processor system.

While a developer is writing a product (source codes, manuals, web pages, etc), a developer may want to search email archive first to read related discussion thread are there. If a developer try to fix a bug, a developer chase a code delta and want to find which changes contribute a bug. While each developer communicate each other, they often to use email archive, which usually available via ftp or web, for a playback of previous discussion. A new user who have an installation problem of the software may want to search an email list for new users, find a person who also met the same problem, then send an email to that person. During each task, people who involved to a software are searching and/or reading code repository and/or email archive.

1.2. An Open Issue

Unfortunately, many problems around OSD is complicated, searching code repository and/or email archive doesn't help developers to solve.

For example, a developer X find a bug in a software, which comes from past program changes D by another developer Y. X soon understands that reverting D from source code fixes a problem, but change D is introduced while implementing an important feature of this software, so reverting D means reverting that feature. X want to know why D is there; X contacts Y by email, but no helps since Y is no longer a developer of this software. X tries to search email archive, but there are lots of emails about this new features. X must check all emails, to find out discussions about design discussion of this feature; X wastes lots of time. Same stories can be found in OSD, such as "easy to read a design discussion, but hard to find what is

the result of code changes", "find a bugfix change, but hard to understand why this change actually fix a bug", and so on.

We argue that there is an open issue that in OSD, code repository and email archives exist independently and no relationship between them. The information all developers want to know are there, but too hard to find out for many cases.

In general, and traditional software engineering environment research, will propose a new archive system, which holds both software products and email and provides search/browse/etc features to developers. It may solve traditional software development, but it is difficult to apply same approach to OSD. In OSD, as shown before, developers' environment are vary from each other; someone can't run new tool, and another one insists to change their environment and/or style. This problem should be solved without any impacts to existing OSD environment, and should not change existing archives.

2. CoxR

In this section, we show our OSD activities browser, **CoxR**, to solve the Open Issue mentioned in previous section. At first, we describe our design policy for **CoxR** and show the overview of **CoxR**.

2.1. Design Policy

Our design policies of **CoxR** are as follows:

Be an Add-on tool, not a replacement tool We are tend to stay environments which used to, and do not want to change them. If **CoxR** is a replacement of their tools, they'll not use **CoxR**. To minimize the impact, we set **CoxR** as an add-on tool for current developers. Developers may or may not use **CoxR**, and **CoxR** should not force developers to use.

Imports existing archives as it is Instead of creating yet another code/email archives for OSD, we just import its archive, and make our internal-use-only database. If there are something changed in code/email archives, we fetch the difference and add new entries to our database. We think "keeping in sync" will not be a problem, since **CoxR** will be a browser of past development activities, not of just-in-time activities.

Has a web interface It is easy and most effective approach that creating our proprietary tool or user-interface for **CoxR**. However, there are portability and data size problem. Developers' environment is different each other, so it is hard to establish **CoxR** a tool for all developers over the world. Code and email repository are tend to be large, so some users can't have the whole data. We design **CoxR** a web-based system; user can use their own HTML browser, and central database is enough for all of developers.

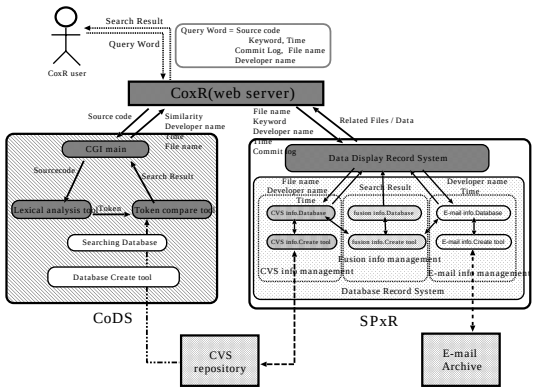

Figure 1. Overview of CoxR

2.2. CoxR Overview

CoxR is consisted of main part integrated with web server, two major subsystem, SPxR[3] and CoDS[4], external CVS repository and email archive (Fig. 1).

SPxR

SPxR (Software Product Xross Reference) analyses both CVS repository and email archive, and calculate a fusion information of both. In SPxR, fusion information connects CVS information (who change which file, when it occurs, and what comment are tied into each change) and email information (who send which mailing list, when it is sent, and what the email sender says), according to same time, same file, and same developers. Intra-CVS and Intra-email connections are also analyzed. SPxR accepts filename, keyword, developer's email address, time, and comment in a commitlog as a search input, then searches not only CVS repository and email archive, but also fusion database. The search result will be a set of filename within the CVS repository, some threads in email archives, etc.

CoDS

CoDS (Code Differences Search) is a search engine for source code itself. Unlike other source code oriented search engine, CoDS has an attention to each change delta of a file, and accepts a code fragment for query string. CoDS employs local algorithm to calculate similarity between input

Figure 2. CoxR entrance page

Figure 3. Similar code search result

code fragment and each change delta of files in CVS repository. If similarity value exceeds pre-defined value, CoDS considers that it is "similar" source code. The search result will be a set of revision number of a filename which is sorted by similarity value, developer name who committed the revision, and time when it was committed.

CoxR main

The main part of **CoxR** is sitting on the web server, and acts as CGI program. **CoxR** handles both SPxR and CoDS subsystem, and accept user's input. When a query word is passed to **CoxR**, **CoxR** passes the input to SPxR and/or CoDS, depends on the input. CoxR also re-query subsystems with their last search result if needed, then send back to the user a query result.

3. System Prototype

We are now developing the **CoxR** browser prototype, using SPxR and CoDS. Figure 2 is the **CoxR** entrance web page. "Revision Search" link guides users to traditional CVS repository and email archive browser. "Keyword Search" accepts simple word, and returns file and email which contains the word. "File Search" is used to shrink search target; **CoxR** try to search all CVS files by default, and it may takes long time to finish searching. "Similar Code Search" provides code fragment search by CoDS. Each search result can be re-used as an input of further search, and if result contains a set of file list it can be used to shrink the target file to be searched.

In following examples, we use FreeBSD CVS repository and its mailing list archive for the original archive. Due to the limit of spaces, we only shows some typical usages.

Figure 4. Revisions matched within a file

3.1. Similar code search

Imagine a developer try to solve a bug. After debugging, a developer find a code fragment which seems to contain a bug itself. However, there are many way to fix and a developer want to know that how this type of bug is fixed by past developers. In **CoxR**, similar code search can be used to search how to fix.

Figure 3 is a result of similar code search. Input code fragment, search status, and matched file name are shown.

The anchors on the filename guides to another page which shows the matched revisions within a file (figure 4.

If user selects arbitrary one revision, CVSweb-like code delta browser are shown, with input code fragment in *italic* font.

CoxR also supports recursive search in general. In figure 6, user can search with a key of yet another code fragment, from files which last matched with the previous code fragment (shown in figure 3).

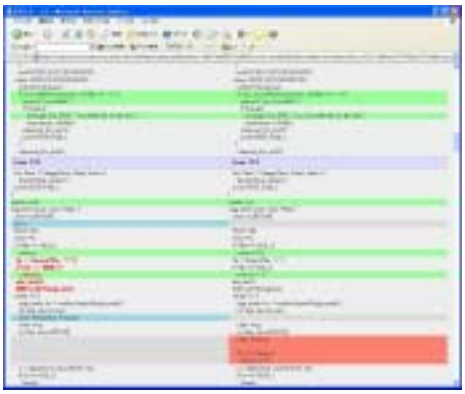

Figure 5. Matched code fragment

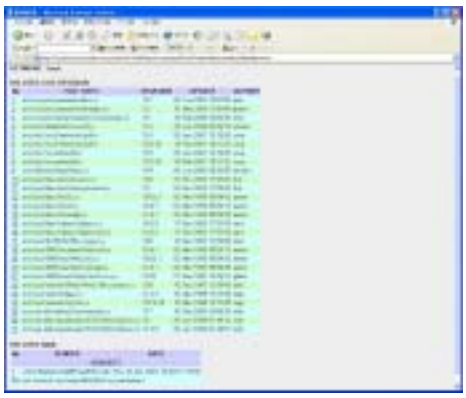

Figure 7. Keyword search result

Figure 6. Recursive similar code search

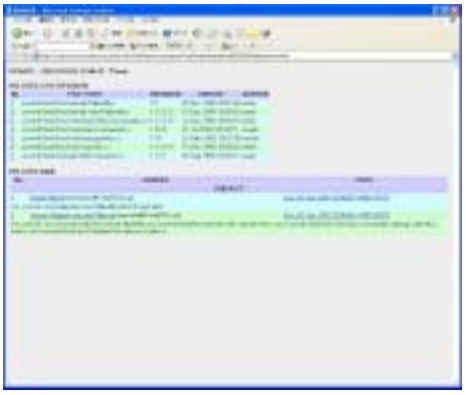

Figure 8. Time search result

3.2. Walking an archive forest

We imagines that both CVS repository and email archive are like a forest; many source codes and emails are there, a series of email discussion and a sequence of code delta and branches become a tree, they are floating each other but binded with time, authors, keywords, filenames, and so on. Understanding OSD activities is considered as walking these forests and finding out what's are there. **CoxR** supports to walk a forest easily.

There are many starting points of your choices including file and/or directories, code fragment mentioned above, a keyword appeared in the source code, etc. In this example, we start with a keyword (fig. 7). CoxR provides both CVS and email result which contains a input keyword.

If a developer interested in the time scale, click a date string. **CoxR** shows source codes and emails which matches the time frame, and other associated files found in fusion database of SPxR (figure 8).

It's easy to chase a specified developer's activity, just click a username and **CoxR** shows the developer's emails and committed revisions (fig. 9, push "CVS" button to show files committed by this developer).

If a filename itself is selected, **CoxR** acts as CVS repository browser (fig. 10). Note that there are many anchors in revisions, time, developer, etc; clicking them makes yet another search with various viewpoints.

If search result contains a file list, similar code search shown in previous example can be performed against to that list.

4. Conclusion

In this paper, we propose our OSD activities browser **CoxR**, and demonstrates with current prototype. **CoxR** guides open-source software developers to CVS repository and email archive forests, and help to understand what activities are performed in past development.

As a further works, **CoxR** usability and performance evaluation should be required. Current prototype only supports list-style of result data, however it can be shown as

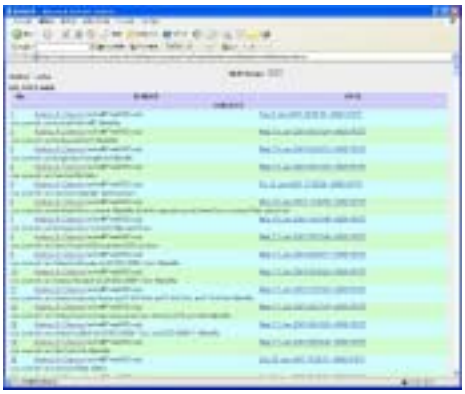

Figure 9. Author search result

Figure 10. Revision history

multi-dimension space with file, developer, time axis and so on.

References

[1] Asklund, U., Magnusson, B., and Persson, A.: “Experiences: Distributed Development and Software Configuration Management”, In Proceedings of SCM9, LNCS-1675, pp.17-33 (1999).

[2] Estublier, J.: “Software Configuration Management: A Roadmap”, The Future of Software Engineering in 22nd ICSE, pp.281–289 (2000).

[3] Ishikawa, T., Yamamoto, T., Matushita, M., and Inoue, K.: “Design of Communication Supporting System with Revision Control System”, IPSJ Technical Report, 2001-SE-133, pp.23–30 (2001).

[4] Tahara, Y., Matushita, M., and Inoue, K.: “Supporting Method for Source Code Modification with the Changes of Existing Software”, IPSJ Technical Report, 2002-SE-136, pp. 57–64 (2002).

[5] Raymond, E.S.: “The Cathedral and the Bazaar”, O’Reilly (1999).