International Journal of Advanced Technology and Innovative Research
www.ijatir.org
ISSN 2348–2370
Vol.07,Issue.17,
November-2015,
Pages:3370-3373

Speculative Analysis of Communications in Code Version Conflicts

R. LAKSHMI TRIVENI
PG Scholar, JNTU, Kakinada, AP, India, E-mail: ratakondatrivenit6@gmail.com.

Abstract: Conflicts among programmers' inconsistent copies of an shared task appear in collaborative development and can decrease progression and quality of shared project. Identifying such situation early can offer assistance. Identifying conditions which may lead to conflict can prevent a few of them. Crystal, a publicly available tool was designed and implemented that unobtrusively provides information about the existence of conflicts in an ongoing and precise way. It uses speculative analysis and mainly focus on qualitative approach. In this we propose a framework to improve backup group functions over rule database so that it upholds quantitative approach.

Keywords: Qualitative And Quantitative, Conflicts And Risks, Crystal Tool Operations, Collaborative Development.

I. INTRODUCTION

Each participant of a collaborative growth venture works on an individual copy duplicate of the venture documents. Each designer continuously makes changes to his or her local duplicate of the documents, shares those changes with the group, and features changes from group members. The reduce synchronization of these actions allows fast growth improvement, but also allows the designers to create multiple, unreliable changes. Disputes can be textual or higher order. A textual dispute arises when two designers create unreliable changes to the same part of the resource rule. To avoid following changes from overwriting past ones, a version control system (VCS) allows the first designer to post changes, but stops the second designer from posting until the issue is settled instantly (by the VCS) or personally (by a developer). Higher order disputes cause collection mistakes, test problems, or other problems, and are challenging to identify and take care of in exercise[3]. As with mistakes in applications, it is generally easier and less expensive to recognize and fix conflicts early, before they distribute in the rule and the appropriate changes disappear in the remembrances of the designers [2].

A. Conflict

Inconsistent copies of shared project in collaborative development may lead to conflict [3][5][8] as shown in Fig.1.

Ex: developer 1 create file x
developer 2 modifies developer 1 file x'
if developer 1 access file x then,
conflict occurs as x is different from x'.

B. Conflict Management

It is the process of limiting the negative aspects of conflicts [9][10][12] while increasing the positive aspects of the conflict. Identifying the situation which may lead to conflicts can prevent some conflict.

Ex: Second developer must wait until first developer publish changes to avoid conflict.

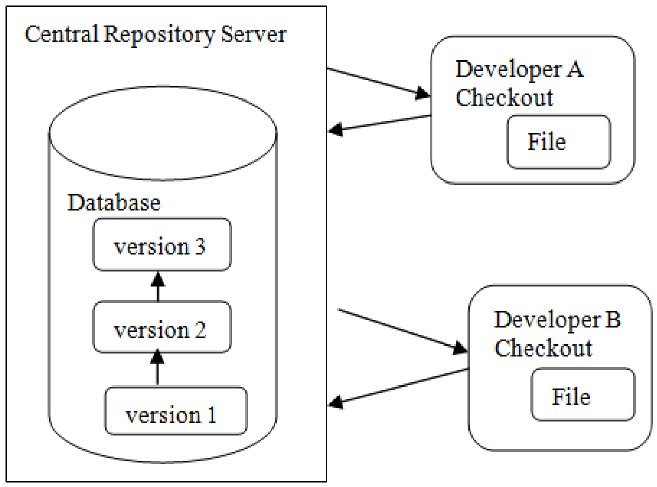


Fig.1. Sharing information from one user to another user.

C. Risk Management Process

Risk: It is a potential prospect damage that may arise from some present events such as a cost swamped. Due to conflicts organizations has to face many risks. They are

- Cost increases
- Time delay

Risk management is a sequence of steps whose objectives are to discover, address, and remove software risks things before they become terrorization.

Risk Management Process Steps:

Step 1: Risk Identification

It allows individual to identify risks, so that the operational staff becomes aware of potential problems. Not only should risk identification be undertaken as early as possible, but it also should be repeated frequently.

Step 2: Risk Impact Assessment

It assess the probabilities and consequences of risk events. The consequences may include cost, schedule, technical performance impact, as well as capability or functionality [4] impacts.

Step 3: Risk Prioritization Analysis

Risk analysis transforms the estimated data about specific risks that developed during risk identification in to a consistent form that can be used to make decisions around prioritization. Risk prioritization enables operations to commit resources to manage the most important risks.

Step 4: Risk Planning & Schedule

Risk planning takes the information obtained from risk analysis and uses it to formulate strategies, plans, change requests and actions. Risk scheduling ensures that these plans are approved and then incorporated into the standard day-to-day process and infrastructure.

Step 5: Risk Tracking

Risk tracking monitors the status of specific risks and the progress in their respective action plans as shown in Fig.2. Risk tracking also includes monitoring the probability, impact, exposure and other measure of risk for changes that could alter priority or risk plans and ultimately the availability of the service.

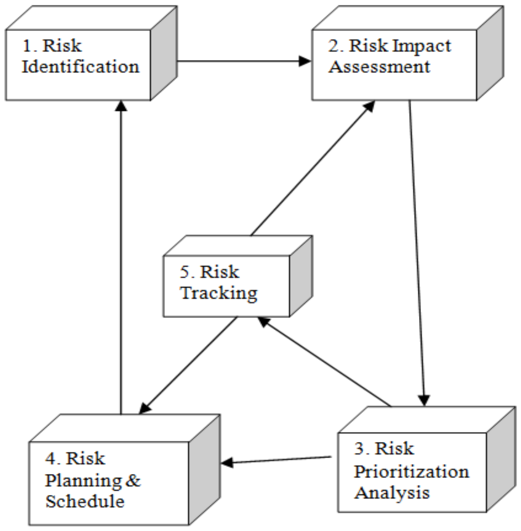
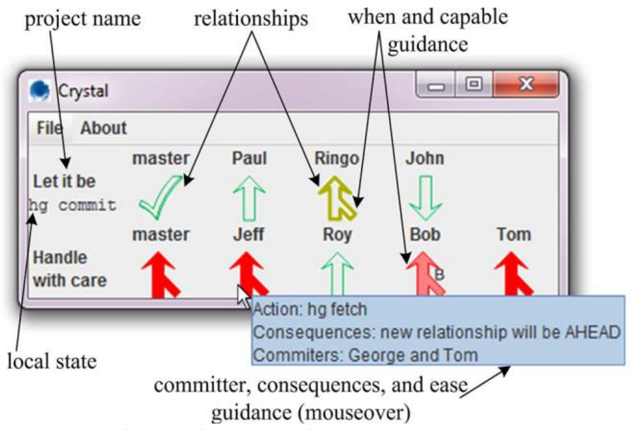

1. Risk Identification
2. Risk Impact Assessment
3. Risk Prioritization Analysis
4. Risk Planning & Schedule
5. Risk Tracking

Fig.2. Risk management process.

II. BACKGROUND APPROACH

Crystal tool, provides the key details without frustrating or annoying the designer in three ways [1]. First, a taskbar symbol in the program plate reviews the most serious state for all monitored databases. A screenshot of george's view of crystal. George is following two projects under development: "Let it be" and "Handle with care". The former has four observed collaborations: George, Paul, Ringo and John; the latter has five: George, Jeff, Roy, Bob and Tom. Crystal shoes George local state and his relationship with the master repository and the other collaborators, as well as guidance based on that information as shown in Fig.3. A designer who likes to get restricted but details need never open the primary screen. (Crystal never reveals any screen asynchronously). Second, the primary screen compactly summarizes all tasks and connections, enabling a designer to immediately check out it to recognize circumstances that may require attention.



Action: hg fetch
Consequences: new relationship will be AHEAD
Commiters: George and Tom

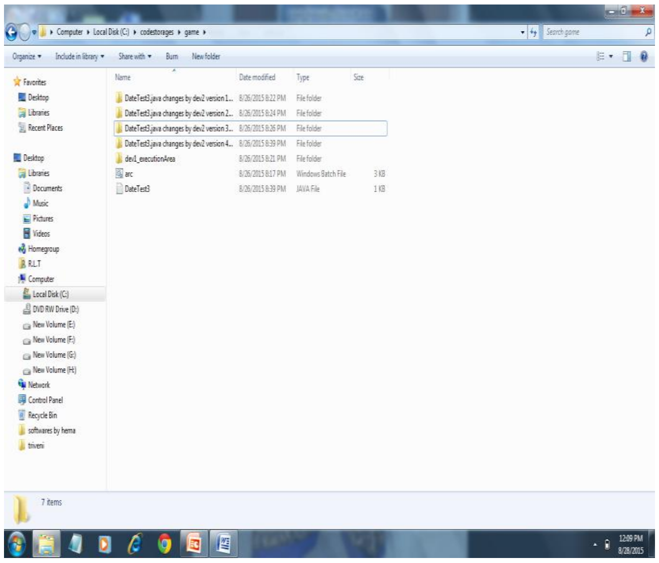
Fig.3.Crystal sample screenshot.

The primary screen reveals symbol taking advantage of shade redundantly and in constant places (rather than, say, a textual list that a designer would have to read and interpret). Each icon's set full, details about each connection, action and assistance is available but invisible until a designer reveals specific interest in it. When the designer moves over an symbol, a tooltip reveals all the details.

III. PROPOSED APPROACH

Bugs or features of the software are often only present in certain versions (because of the fixing of some problems and the introduction of others as the program develops). Evaluation of crystal tool is preliminary and qualitative. Efforts needed to be develop to evaluate it via both qualitative and quantitative approaches. Crystal's automation framework replace many of the existing command mode operations is efficient to support many project relationships. We propose to extend its utility by enhancing it to support batch operations over code repositories so that it upholds the quantitative approach. Bulk operations over code storages enhance the crystal tool further and a prototype validates our claim. In this approach first project are assigned to developer depending on their area of development. After assigning complete project details are stored in database which can viewed by only who can be logged in to the database. The project details include project category, user name, name of the file developed by that user, path where the file is stored, version and uploaded date and time. It provides the information about the original developer of the file. In collaborative development developers work together on same project to complete their task. In this development method any developer can access any file at any instant and can able to make changes to that file which was not originally developed by that user.

In existing approach changes made by each developer is not saved only recently made changes are only saved. But in proposed approach modifications done on a file by different users are saved using a framework. Backup is provided for each operation done on file. A framework is developed in such a way that complete details are stored in code storages folder in drive which can be accessed by authorized developers. Here files are stored according to project category. Files that are developed for same project by different users are also stored in code storages based on project category. In this approach modified file is stored with file name including user name and version as shown in Fig.4. Here user name is the name of the user who committed the file and version number varies depends on number of times modifications done on that file. By providing backup for batch operations can upholds quantitative approach.


Computer ▸ Local Disk (C:) ▸ codestorages ▸ game ▸

Name	Date modified	Type	Size
DateTest3.java changes by dev2 version 1...	8/26/2015 8:22 PM	File folder	
DateTest3.java changes by dev2 version 2...	8/26/2015 8:24 PM	File folder	
DateTest3.java changes by dev2 version 3...	8/26/2015 8:26 PM	File folder	
DateTest3.java changes by dev2 version 4...	8/26/2015 8:39 PM	File folder	
dev1_executionArea	8/26/2015 8:21 PM	File folder	
arc	8/26/2015 8:17 PM	Windows Batch File	3 KB
DateTest3	8/26/2015 8:39 PM	JAVA File	1 KB

7 items

Fig.4. A screenshot which shows the backup files that are saved in drive.

IV. COMPARISSION

We compared the enhanced method with previous work. We recommended to improve its application by improving it to backup group functions over program code databases so that it upholds quantitative parameter. The approach used in previous work will focus mainly on quality of data with no backup.

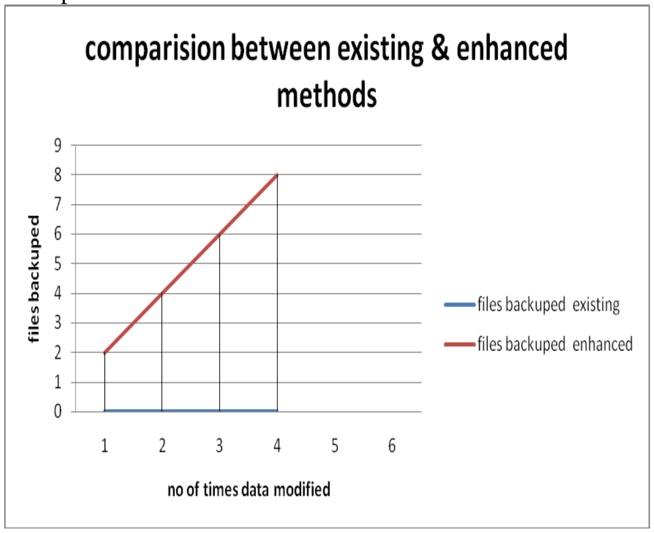


Fig.5. A sample graph which differs existing and enhanced work.

The enhanced method focus on backup. Our experimental results show efficient process communication event management with preferred program evaluation. In existing system only recently modified data can only be accessed but in enhancement all the modified data is stored in a drive which can be accessed by any one at any time with this modification data can be easily identified as shown in Fig.5. The graph shows the difference between proposed and enhanced work by providing backup facility.

V. CONCLUSION

Speculative analysis over version control operations provides precise information about pending conflicts between collaborating team members. These pending conflicts including textual, build and test are guaranteed to occur. Learning about them earlier allows developer to make better informed decisions about how to proceed, whether it is to perform a safe merge, to publish a safe change, to quickly address a new conflict, to interact with another developer and so in. NCSL indicates that

- Conflicts are norm rather than the exception,
- 16 percent of all merges required human effort to resolve textual conflicts,
- 33 percent of merges that were reported to contain no textual conflicts by the VCS in fact contained higher order conflicts.

To date, developments in software programs for analysis of qualitative data that have contributed most noticeably to researchers capacity. Initiatives required to be designed to assess it via both qualitative and quantitative factors. Crystal's automated framework to substitute many of the current control method functions is effective to back up many venture connections. To improve its application by improving it to back up group functions over program code database so that it upholds quantitative parameter.

VI. REFERENCES

[1]Yuriy Brun, Reid Holmes "Early Detection of Collaboration Conflicts and Risks ," ieee transactions on software engineering. vol 39, no 10,OCT 2013.

[2] P . Dewan, "Dimensions of Tools for Detecting Software Conflicts," Proc. Int'l Workshop Recommendation Systems for Software Eng, pp. 21-25,Nov. 2003.

[3].S. Horwitz J. Prins, and T. Reps, "Integrating Noninterfering Versions of Programs." ACM Trans. Programming Languages and Systems, vol 11, pp. 345-387,july 1989.

[4] . Hemalatha Narne, Adepu Sridhar, "Speculative Analysis Exploits Qualitative and Quantitative User Studies," IJDCST Feb-March Issue V-2, I-3, SW-06.

[5]. C.R.B. de Souza, D. Redmiles and P. Dourish, "Breaking the Code,' Moving between Private and Public Work in Collaborative Software Development." Proc. Int'l ACM SIGGROUP Conf. Supporting Group Work, pp.105-114, Nov. 2003.

[6]. C. Bird and T. Zimmermann, “Assessing the Value of Branches with What-If Analysis,” Proc. ACM SIGSOFT 20th Int’l Symp. Foundations of Software Eng., 2012.

[7]. Y. Brun, R. Holmes, M.D. Ernst, and D. Notkin, “Speculative Analysis: Exploring Future States of Software,” Proc. FSE/SDP Workshop Future of Software Eng. Research, pp. 59-63, Nov. 2010.

[8]. J. Estublier and S. Garcia, "Process Model and Awareness in SCM," Proc. 12th Int'l Workshop Software Configuration Management, pp. 59-74, Sept 2005.

[9]. R.E. Grinter, "Using a Configuration Management Tool to Coordinate Software Development," Proc. Conf. Organizational Computing Systems, pp. 168-177, Aug.1995.

[10]. D.E Perry, H.P. Siy, and L.G. Votta, " Parallel Changes in Large-Scale Software Development: An Observational Case Study," ACM Trans Software Eng. and Methodology, vol.10, pp.308-337, July 2001.

[11] . J. D. Knowles and D. Corne. Properties of an adaptive archiving algorithm for storing non dominated vectors.
IEEE Transactions Evolutionary Computation, 7(2):100–116, 2003.

[12]. T. Zimmermann, "Mining Workshop Updates inCVS," Proc. Fourth Int'l Workshop Mining Software Repositories, May 2007.

[13]. M. L´opez-Ib´a˜nez, J. Knowles, and M. Laumanns. On sequential online archiving of objective vectors. In R. H. C. Takahashi, K. Deb, E. F. Wanner, and S. Greco, editors, Proceedings of Evolutionary Multi-criterion Optimization (EMO 2011), volume 6576 of Lecture Notes in Computer Science, pages 46–60. Springer, 2011.

[14]. F. Neumann and J. Reichel. Approximating minimum multicuts by evolutionary multi-objective algorithms. In Proceedings of Parallel Problem Solving from Nature X (PPSN ’08), volume 5199 of Lecture Notes in Computer Science, pages 72–81. Springer, 2008.