Automatic Version Control System for Distributed Software Development

PROJECT PLAN

Project Type:	Master thesis
Project Period:	26.03.2012 – 26.09.2012
Student:	Sandra Weber
Status:	4th Semester Computer Science at ETH
Email address:	sweber@student.ethz.ch
Supervisor:	Martin Nordio, H.-Christian Estler and Prof. Bertrand Meyer

1. PROJECT DESCRIPTION

Overview

Software development nowadays is global. Companies cross borders in the search of qualified personal. Teams are spread over more than one countries and have to deal with the difficulties of different time zones and other cultures [3, 5, 6, 8]. The distributed collaboration results into many new challenges that must be faced like communication, requirements engineering and API design [11]. The difficulties of distributed software engineering have been researched by M. Nordio et al. on the example of the DOSE course [7, 10].

Each software developer knows this situation: You are working at a team of two or three people. Everybody has his own computer, yet you all end up looking at the same screen. Even if each person keeps working on their own laptop, before being able to share the newly added code the annoying circle of software configuration management "commit-resolve-update" must be completed, before the other team members can comment on the new code. Nowadays interactive programming techniques like pair programming or code review are commonly used. Yet the standard IDE offers little support for this kind of collaboration.

The software development environments (IDEs) are the most important tool for a developer. But even so we work in teams, an IDE is still a personal tool. Cloudstudio moves the IDE to the cloud [9]; this does not just follow the trend to offer services in the cloud, but reacts to the needs of distributed software development. Cloudstudio offers the possibility to simultaneously program on different software elements, to validate code using its verification framework and to communicate using integrated tools. Another main feature of Cloudstudio is the simplification of the software configuration management, which will be further improved during this project. The user no longer has to worry about committing and sharing the newly added code; the others can directly see his live changes. As soon as the code compiles correctly it will be automatically committed; there is no worrying about committing and updating to be up-to-date with recent changes of other team members. The goal of this project is to improve the

software configuration management of Cloudstudio such that the user has all the comfort of an automatic version control system, while still having additional, useful features of standard version control systems at hand.

Scope of the work

During this project the version control of Cloudstudio will be redesign. The redesign will solve the issue of user A receiving compilation errors because of what user B is typing, while still providing live code collaboration and automatic committing. The already existing features, like manual commit or rollback, will still exist in the new system, but due to the redesign accessing the project code from external development environments like Eiffelstudio or Eclipse will be possible.

In the next step further additional features will be added e.g. enabling the user to go to a specific revision or to look at the revision log. As an option, a tool will be developed to work on a Cloudstudio project with another IDE; and a case study will be performed to test and evaluate the new features of Cloudstudio.

Intended results

The result of this project will be an improved version control system for Cloudstudio. The new system will support the typical features of version control like commit and rollback. Further the code entered in Cloudstudio will be automatically committed after compiling, if no compilation errors occurred and the other users can see the code entered live.

It will be also possible to edit code in a Cloudstudio project using another IDE, like Eiffelstudio or Eclipse. The extern user will not be able to see live changes of other users, but he can commit code and the changes will be visible for users of Cloudstudio.

2. BACKGROUND MATERIAL

Reading list

[1] Git Documentation. http://git-scm.com/documentation, April 2012.

[2] Git Community Book. http://book.git-scm.com/index.html, April 2012.

[3] Mercurial Guide. http://mercurial.selenic.com/guide/, April 2012.

[4] Marcel Bertsch. A web-based IDE for Java. Semester project of the Software Engineering Laboratory, ETH Zurich, 2011.

[5] Alexandru Ioan Dima. Developing JavaScript applications in Eiffel. Master thesis, ETH Zurich, 2011.

[6] Roland Meyer. Revision control support for a web-based IDE. Semester project of the Software Engineering Laboratory, ETH Zurich, 2012.

3. PROJECT MANAGEMENT

Objectives and priorities

- New design of a version control system for Cloudstudio
- Integration of the version control system into Cloudstudio
- (optional) Tool to commit from the outside
- (optional) Case study to test and evaluate the new version control system
- Results documented in the master thesis report

Criteria for success

The minimal quality requirements[1] are the following:

- The version control system supports manual commits from an extern source.
- A user of Cloudstudio can see changes committed by an extern source.
- An user of Cloudstudio can activate an according option to see live changes of other users of Cloudstudio.
- The live changes of other users do not inflict compiling errors.
- The code of a Cloudstudio user is automatically committed after compiling, if no compiling errors occurred.
- The user can manually commit all current changes.
- The manually committed changes are visible for an extern source.
- The user of Cloudstudio can go back to the previous commit (rollback).

The thesis meats the expectation if at least three of the following five points are successfully implemented:

- When the Cloudstudio user decides to push the changes, he can specify a message.
- The user of Cloudstudio can go back to any previous revision.
- The user of Cloudstudio may select which files are pushed.
- When pushing or committing results in a conflict, the user is offered a possibility to resolve those conflicts
- The user can consult a log of the latest pushes of all the users.

The result is outstanding and exceeds the expectation, if all the points listed before are fulfilled and the following criteria apply:

- A tool is provided to push for an extern source (e.g. Eclipse or Eiffelstudio).
- A case study with approximately six persons is performed, during which the Cloudstudio with the new version control system is tested and evaluated.

---

[1] According to the Computer Science Department regulations, a thesis that meets the minimum quality requirements should be graded with 4.0.

Method of work

See project steps below.

Quality management

Documentation

The master thesis is documented in the report.

Validation steps

- The current status will be discussed in a weekly meeting.
- Theoretical scenarios are run continuously during development to test the functionality (Test cases).
- JUnit tests are written, if it is possible and reasonable.
- (optional) During a case study similar to the one performed in [1], Cloudstudio with automatic version control is tested by several test users.

4. PLAN WITH MILESTONES

Project steps

Milestones:

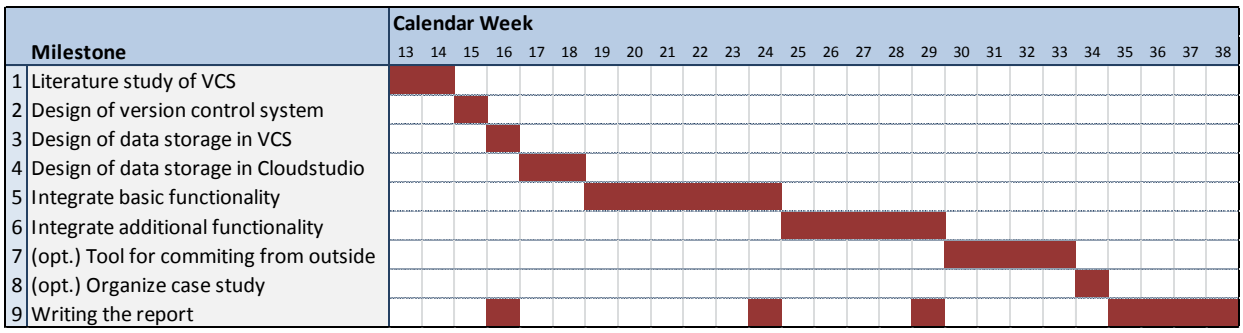
1. Literature study of Version Control systems (Mercurial, Git, etc). Analyze their functionality. Get familiarized with Cloudstudio.
2. Design of the version control system
3. Design of the storage of the data in the version control system
4. Design the storage of the data in Cloudstudio
5. Integrate basic functionality (commit, push, rollback) into Cloudstudio
6. Integrate additional functionality (log, go to revision, resolve conflicts, ..) into Cloudstudio
7. *) Create a tool to commit from the outside without the distributed IDE
8. *) Organize a case study with several persons
9. Write the master thesis report

*) This milestone is optional. It is part of the criteria of an outstanding result.

Deadline

Wednesday, 26.09.2012 (Week 39)

Tentative schedule



REFERENCES

[1] Bertrand Meyer: *Object-Oriented Software Construction, 2nd edition*, Prentice Hall, 1997.

[2] Chair of Software Engineering: *Semester-/Diplomarbeiten*; Online at: http://se.inf.ethz.ch/student_projects/, consulted in March 2012.

[3] E. Carmel. *Global software teams: collaborating across borders and time zones*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1999.

[4] H.-C. Estler, M. Nordio, C. A. Furia, B. Meyer, and J. Schneider. Agile vs. structured distributed software development: A case study, 2012. ETH technical report.

[5] J. A. Espinosa, N. Nan, and E. Carmel. Do gradations of time zone separation make a difference in performance? a first laboratory study. *In Proceedings of the IEEE International Conference on Global Software Engineering (ICGSE 2007)*, pages 12–22. IEEE, Aug. 2007.

[6] J. D. Herbsleb, A. Mockus, T. A. Finholt, and R. E. Grinter. Distance, dependencies, and delay in a global collaboration. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, CSCW '00, pages 319–328, New York, NY, USA, 2000. ACM.

[7] M.Nordio, C.Ghezzi, B.Meyer, E.D.Nitto, G.Tamburrelli, J.Tschannen, N.Aguirre, and V.Kulkarni. Teaching software engineering using globally distributed projects: the DOSE course. In *Collaborative Teaching of Globally Distributed Software Development - Community Building Workshop (CTGDSD)*, New York, USA, 2011. ACM.

[8] M. Nordio, H.-C. Estler, B. Meyer, J. Tschannen, C. Ghezzi, and E. D. Nitto. How do distribution and time zones affect software development? a case study on communication. In *Proceedings of the IEEE International Conference on Global Software Engineering (ICGSE 2011)*. IEEE, 2011.

[9] M. Nordio, H.-C. Estler, C. A. Furia, and B. Meyer. Collaborative software development on the web, 2011. arXiv:1105.0768v3.

[10] M. Nordio, R. Mitin, and B. Meyer. Advanced hands-on training for distributed and outsourced software engineering. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering (ICSE)*, 2010.

[11] M. Nordio, R. Mitin, B. Meyer, C. Ghezzi, E. D. Nitto, and G. Tamburrelli. The role of contracts in distributed development. In *Proceedings of Software Engineering Approaches for Offshore and Outsourced Development*, 2009.