The impact of requirements in distributed software development: an empirical study

PROJECT PLAN

Project type:	Master's Thesis
Project period:	December 1st, 2014 - June 1st, 2015
Student name:	Marc Egg
Status:	4th Semester, Master of Science ETH in Computer Science
Email address:	megg@student.ethz.ch
Supervisor name:	Prof. Dr. Bertrand Meyer, Chair of Software Engineering, ETH Zurich
	Dr. Martin Nordio, Chair of Software Engineering, ETH Zurich

1. PROJECT DESCRIPTION

Overview

Distributed software development is the development of software products by multiple development teams. It gave rise to various new problems compared to traditional software development where one development team produced a software product. The challenges of distributed software development include among others more complicated communication among multiple developers and their teams, different time zones and thus different and not necessarily overlapping office hours, language barriers and various cultural backgrounds.

These challenges require the traditional software processes used to develop software products to be adapted or replaced to reflect the differences introduced by the distributed project setting. Distributed software development raises interesting questions for software engineering researchers. One particularly interesting question is how requirements influence the quality of the software product being developed in a distributed project setting.

This project, a master's thesis submitted for the degree of *Master of Science ETH in Computer Science* (*MSc ETH CS*) plans to show how requirements affect the quality of software products developed in a distributed project setting. The data set used originates from the *ETH Zurich* course *Distributed Software Engineering Laboratory* (*DSL*). Until 2012 the course's name was *Distributed and Outsourced Software Engineering* (*DOSE*). The *Chair of Software Engineering* [CSE] at *ETH Zurich* hosts *DSL*, an university-level course for computer science students, every fall semester. The course aims at educating the participating students in distributed and outsourced software engineering. To do so *DSL* is accompanied by a mandatory distributed software development project which includes both writing a requirements document in the spirit of *IEEE Std 830-1998* [IEEE] and implementing the software product as defined by the requirements. Participating universities include universities from Europe, South America, Africa, Asia and Australia.

The master's thesis, an empirical study, uses the deliverables from the DSL course as a data set to research the impact of requirements in distributed software development.

Scope of the work

This project tries to establish relationships between the quality of requirements documents and the quality of software products. To do so, the requirements documents must be read and their quality must be judged according to some grading scheme. The grading scheme to be used is one which was developed for another course thaught by the members of the *Chair of Software Engineering* that also includes writing a requirements document in the spirit of IEEE Std 830-1998 [IEEE] as part of the mandatory project work.

The next step is to grade the software products using *Eiffel Inspector* [EIFINSP]. *Eiffel Inspector* is a static code analysis tool that finds problems in the implementation of a software products. Note that the notion of *problem* does not relate to a bug but to a problematic implementation pattern.

Once both requirements documents and software products are graded statistical tests are used to find relationships between the quality of requirements documents and the quality of software products.

The last part of this project is to formally write down the findings into a report which also constitutes the master's thesis.

Intended results

The results of this project are statements how the quality of requirements documents affects the quality of software products. The report has to discuss the reasons if no statement can be made or no relationship can be found.

2. BACKGROUND MATERIAL

Reading list

- DSL course setting
    - 2008: [DOSE08]
    - 2009: [DOSE09]
    - 2010: [DOSE10]
    - 2011: [DOSE11]
    - 2012: [DOSE12]
    - 2013: [DSL13]
    - 2014: [DSL14]
- Requirements
    - DSL (2014) lecture slides [REQS]
    - IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998) [IEEE]
    - Grading scheme for requirements documents (not publicly available)
- Software qualities
    - Software Architecture (2011) lecture slides: [ARCH]
    - Eiffel Inspector: [EIFINSP]

3. PROJECT MANAGEMENT

Criteria for success

The following quality requirements assume that all requirements documents exist and that all software products successfully compile without or with minor changes to the source code.

Minimum quality requirements

- Grading both requirements documents and software products of one to 24 groups (years 2008 - 2012).
- Basic statistical tests.

Expected requirements

- Grading both requirements documents and software products of 25 to 29 groups (years 2008 - 2012).
- Statistical tests.

Requirements to exceed the expected requirements

- Grading both requirements documents and software products of 60 to 71 groups (years 2008 - 2013).
- Grading the software product from an user's point of view.
- Statistical tests.

Method of work

See project steps below.

Quality management

Documentation

A written report documents the master's thesis.

Validation steps

Weekly meetings where the progress and work of one week are discussed ensure the project progress with the necessary pace to guarantee a successful and timely completion of the project.

4. PLAN WITH MILESTONES

Project steps

Milestones are based on the following tasks:

1. Organizing and structuring the unorganized DSL repositories starting with year 2008 until 2013 (inclusive) such that the composition of teams, the choices of subcomponents of teams, existence of documents, source code and videos are known. Once completed potential problems such as missing deliverables and group reorganizations have to be identified and potential exclusion of groups from the project have to be discussed.
2. Studying the lecture slides about requirements and the used standard to refresh the knowledge about requirements. This includes reading the grading scheme for requirements documents as well.
3. Grading the requirements documents according to the given grading scheme.
4. Developing a grading scheme given the set of rules Eiffel Inspector uses to find problems related to the source code.
5. Running Eiffel Inspector on the source code and gathering the results. This includes the application of the previously developed grading scheme.
6. (optional) Developing a grading scheme to judge the quality of the software products from an end user's point of view.
7. (optional) Evaluating the quality of the software products using the previously developed grading scheme.
8. Applying statistical tests to find relationships between quality attributes of a requirements document and quality attributes of a software product.
9. Writing the master's thesis
10. Preparing the presentation.

Deadline

June 1st, 2015 (Week 23, 2015)

Tentative schedule

Milestone	Completion by the end of
1	week 49, 2014
2	week 49, 2014
3	week 3, 2015
4	week 4, 2015
5	week 7, 2015
6 (optional)	week 8, 2015
7 (optional)	week 10, 2015
8	week 15, 2015
9	week 19, 2015
10	week 19, 2015

References

[ARCH] Bertrand Meyer et al.: Lecture 6: Designing for reuse, Software Architecture, 2011, http://se.inf.ethz.ch/courses/2011a_spring/soft_arch/lectures/06_softarch_design_for_reuse.pdf, accessed December 2014

[CSE] Chair of Software Engineering, http://se.inf.ethz.ch/, accessed December 2014

[DOSE08] Distributed and Outsourced Software Engineering, 2008, http://se.inf.ethz.ch/old/teaching/2008-H/dose-0273/, accessed December 2014

[DOSE09] Distributed and Outsourced Software Engineering, 2009, http://se.inf.ethz.ch/old/teaching/2009-H/dose-0273/, accessed December 2014

[DOSE10] Distributed and Outsourced Software Engineering, 2010, http://se.inf.ethz.ch/courses/2010b_fall/dose/, accessed December 2014

[DOSE11] Distributed and Outsourced Software Engineering, 2011, http://se.inf.ethz.ch/courses/2011b_fall/dose/, accessed December 2014

[DOSE12] Distributed and Outsourced Software Engineering, 2012, http://se.inf.ethz.ch/courses/2012b_fall/dose/, accessed December 2014

[DSL13] Distributed Software Engineering Laboratory, 2013, http://se.inf.ethz.ch/courses/2013b_fall/dsl/, accessed December 2014

[DSL14] Distributed Software Engineering Laboratory, 2014, http://se.inf.ethz.ch/courses/2014b_fall/dsl/, accessed December 2014

[EIFINSP] Stefan Zurfluh: Rule-based code analysis, Master's Thesis, 2014, http://se.inf.ethz.ch/student_projects/stefan_zurfluh/report.pdf, accessed December 2014

[IEEE] Institute of Electrical and Electronics Engineers: IEEE Recommended Practice for Software Requirements Specifications, IEEE Std 830-1998, 1998, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=720574, accessed December 2014

[REQS] Bertrand Meyer et al.: Requirements Analysis, Distributed Software Engineering Laboratory, 2014, http://se.inf.ethz.ch/courses/2014b_fall/dsl/slides/04-requirements.pdf, accessed December 2014