Saving a Runaway Project
(a case study for project turnaround)

Yury Makedonov, CGI

International Quality Conference, Toronto, October 05, 2005

Agenda

- Introduction of the case study
- Challenges of this project.
- Description of the actual process that was used for the successful completion of this project. The following critical success factors will be discussed:
  - Active involvement of the project sponsor, fast decision making.
  - Active user involvement, instant feedback
  - Co-location, effective communication
  - Iterative and incremental development in a customer selected order.
- Comparison of this process to current agile thinking.

Story of a real life software project

- It was a new application for a sales department (~100 people) of a division of a big company.
- The goal of this department was to configure and sell telecommunication products.
- The new application was to replace an existing legacy Client/Server application.
- The modern web application was to have a sexy web interface and more functionality.
- This department did not have an established software acquisition/implementation process.
- The vendor was a start-up company trying to establish itself. 10-15 developers were working on this project initially.

History

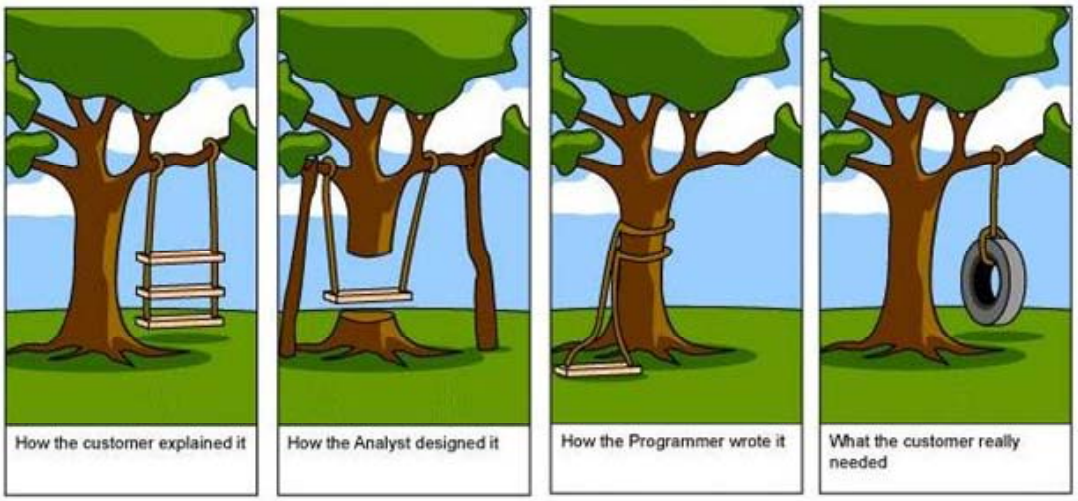
This project followed a classic review and sign-off approach:

- Requirements were reviewed and signed off
- Design of the system was reviewed and signed off
- Specifications were reviewed and signed off
- Development was finished and tested by a vendor

The first attempt to deliver the application to the customer failed. Customer's reaction was:

- "This is not what we wanted!"

Customer: “This is not what we wanted!”



- How was this possible despite the rigorous process of walkthroughs and sign-offs?

Why did this happen?

- The following assumptions are paramount to the success of the review process:
  - All steps are relatively defect free.
  - Efficiency of defect detection by these reviews is high.
  - Defect removal efficiency is high.
- Unfortunately these assumptions are not valid 100% of the time.

Root cause of this problem

Let's take a look at what really happened:

- The Business Analyst (BA) talked to users and stakeholders.
- Users and stakeholders explained what they needed.
- Business Analyst got a **perception** that he understood these users.

Root cause of this problem

In reality the users and the Business Analyst were speaking different languages:

- they used different meanings of the same English words,
- considered different contexts,
- made different assumptions.

As a result the Business Analyst significantly misinterpreted the application users' needs and created a flawed requirements document.

Actual review and sign-off process

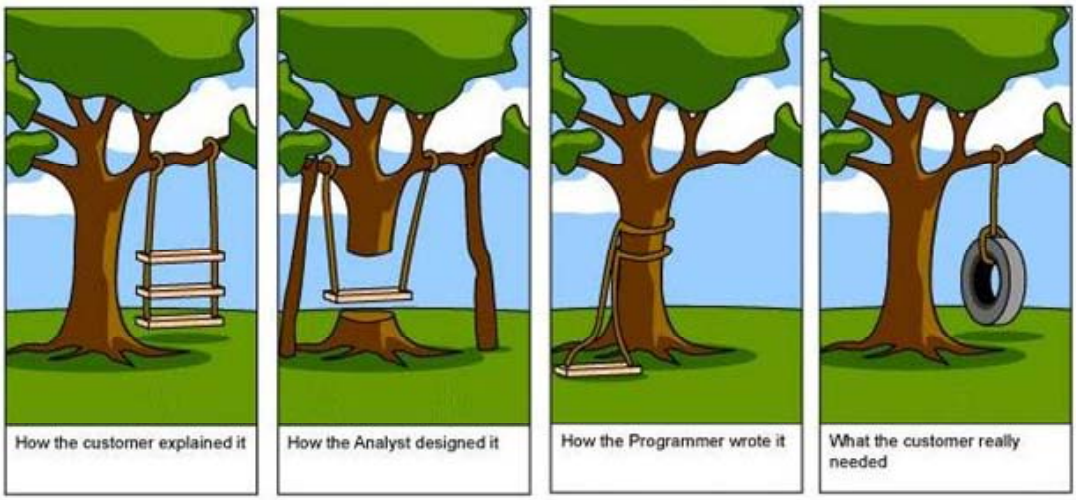
- This requirements document was then sent back to the users for review:
  - Business users got a **perception** that they understood the requirements document.
  - They provided some comments and, after their concerns were addressed, they signed off on the requirements document.

Actual review and sign-off process

- But remember, the business users and the Business Analyst were speaking different languages!
- So these modifications were irrelevant to the quality of the requirements document and major flaws were missed by the reviewers.
- We see that in this case the review/sign off process was useless and only wasted valuable time.

History

Customer: “This is not what we wanted!”



11 2005 International Quality Conference, Toronto, Ontario, Canada October 5, 2005

Episode II - One more attempt

- The requirements were modified using the same review/sign off approach.
- The vendor implemented these updated requirements.
- To make the acceptance process more organized the customer brought in an external Test Manager to supervise the user acceptance testing.

12 2005 International Quality Conference, Toronto, Ontario, Canada October 5, 2005

Development of acceptance test cases

- End users were brought to the project team to develop acceptance test cases to verify the requirements.
- Some of them worked on commission and special arrangements about their salaries were required.
- They started working together with “professional” testers and were trained on how to develop test cases.
- Users were asked to use their own language when writing these test cases. They ranked all existing requirements and started developing test cases for the most important requirements.

Development of acceptance test cases

- The first version of the test cases was incomprehensible for professional testers and developers.
- The test cases were modified and the second version could be understood by both testers and developers.
- Further on these acceptance test cases came to be used instead of the original requirements.
- These test cases still had many errors but nevertheless were much better than original and revised “requirements.”

Start of testing

- Users started testing as soon as acceptance test cases for the major functions were developed.
- The first round of testing was a complete disaster:
  - Most test cases failed.
  - It was impossible even to initiate execution of many test cases - they depended on successful executions of other test cases.
- Many “Severity 1” and “Severity 2” defects were recorded (high severity defects).

15 2005 International Quality Conference, Toronto, Ontario, Canada October 5, 2005

Status of the project

- It was already the second iteration.
- Apparently we had almost all the right pieces but could not fit the puzzle together!

16 2005 International Quality Conference, Toronto, Ontario, Canada October 5, 2005

Pilot and Release 1

- There was an extensive discussion with the project sponsor, developers and end users (acceptance testers).
- A decision was made to break all functions into two releases:
  - Pilot:
    - Bare minimum functionality - not even all the functions of the legacy application were implemented.
    - Pilot was inferior to the existing legacy application and could not be used in real production.
    - Its goal was to show the release to a wider community of sales people.
  - Release 1:
    - Fully functional product with all remaining features.

Change Control Board established

- To better manage this project a “Change Control Board” was established. It consisted of:
  - the project sponsor,
  - users of the application (acceptance testers) and
  - the developers’ managers.
- The Board approved a list of defects (including several missing functions) to be fixed for the “Pilot” release.
- Only defects of the highest severities (“Severity 1” and “Severity 2”) were to be fixed.

Defects fixing

- Developers were not allowed to fix any defect or implement any new feature without the approval of this board.
- The developers were allowed to work only on defects of the highest priority:
  - defects fixing of wich was essential for the “Pilot” implementation, and
  - defects that prevented the execution of major test cases.
- The repair of not so severe defects was postponed despite the developers’ promises that some of them required “only 10 to 30 minutes to fix.”

Role of end users

- The vendor had daily builds delivered for testing.
- End users (acceptance testers) were available to clarify specific details of test cases (requirements) and defect reports for developers.
- Those clarifications were important because initially the developers had a lot of problems interpreting the test cases and defect reports.
- Development of new acceptance test cases continued.

Role of end users

- Why were end users more efficient at developing and executing test cases?
- They have working knowledge of their business – how to configure a product and create an order.
- They were able to use their knowledge better when creating and executing test cases than when talking to a business analyst and trying to understand his gibberish language.

Release of end users

- Initially four end users were used full time for four weeks to develop these acceptance test cases and to start testing the application.
- After four initial weeks professional testers were able to execute these acceptance test cases, and the four business users were gradually released back to their main jobs.
- Nevertheless, one of these business users, on a rotational basis, was working with testers and developers every day.
- The remaining business users were available for phone consultations. Unfortunately this arrangement was not as good as having them on site. They had their own responsibilities and as a result their response time was not as good as when they were working on site.

Change Control Board role

- New test cases, which were essentially new requirements, were reviewed and approved by the same Change Control Board.
- The Board discussed:
  - Progress of testing,
  - Fixed defects,
  - New discovered defects,
  - Defects reported as fixed by developers that were “failed” by acceptance testers.
- Priority of remaining defects and additional “new” defects were reviewed and revaluated every day.
- Developers were allowed to work only on defects of highest priority in this stack!

Change Control Board

- During the first week, the Control Board was meeting every day under the direction of the project sponsor.
- Initially the project sponsor said that he had no time to attend these daily meetings, but was told that without his involvement he would not get the project implemented.
- Project sponsor involvement allowed all required decisions to be made really fast.
- After the first week the Control Board continued meeting every day.
- The project sponsor chaired it twice a week and then once a week until the “Pilot” was implemented.

Pilot, Release 1 and Release 2

- For two weeks the Pilot was used by a wider community of users concurrently with the legacy application
- After the end of Pilot a decision was made to break “Release 1” into:
  - Release 1:
    - about the same functionality as the legacy application plus a couple of new features.
  - Release 2:
    - remaining features of original “Release 1”.

Release 1

- “Release 1” was developed using about the same approach.
- The frequency of Change Control Board meetings was changed to twice a week.
- The project sponsor was chairing this Change Control Board once a week.
- “Release 1” was finally implemented almost two years after project initiation instead of less than a year as was originally planned.

Release 2

- The scope of “Release 2” was reviewed based on first results of “Release 1” in production.
  - Some features were removed.
  - Other new features were added.
- In four months “Release 2” was developed and implemented using about the same approach.
- For the last 3 years this application was in production with just minor modifications. This project was recognised as a success despite all original problems.

Summary

- What allowed for an efficient turnaround of this project:
  - Active involvement of the project sponsor, fast decision making.
  - Active user involvement, instant feedback
  - Co-location, effective communication
  - Iterative and incremental development in a customer selected order.
- These are:
  - core principles of different modern agile software development methods **and** at the same time
  - common-sense principles that started being used many years ago by successful software managers.
- The same approach is recommended for any project of a similar complexity.

Q & A

- Questions?

Contact information

Yury Makedonov, CSTE, CSQA
Senior Technical Consultant
Centre for Testing and Quality
CGI services to BCE
125 Commerce Valley Drive West, Floor 6B,
Markham, ON, L3T7W4
(416) 481-8685
ivm@ivm-s.com
http://www.softwaretestconsulting.com

Appendix - literature

- You can find more information on agile processes, for example, in:
  - Kevin, J. Aguanno (editor), “Managing Agile Projects“.
  - http://www.agilealliance.org/articles/index

Appendix - Test team organization

How the whole test team was organized:

- Business users were developing “scenario like” test cases which required their intimate knowledge of business processes.
- Professional testers concentrated on classical valid/invalid test cases, boundary conditions, etc.
- There was even one person working on GUI test automation. He focused of automation of “Smoke testing” or “Acceptance into testing” tests - a small set of test cases that was executed as soon as a new build was delivered for testing.