A study on user-centered design illustrated by development of social platform for dance classes attendees

Tomasz Legutko

2016/2017

Contents

1 Motivation

Software development methodologies have always been and always will be an important part of all IT projects. With every year software becomes increasingly important part of our society and with every year the software's complexity becomes even harder to manage. Over the years there's been a great progress in architectural patterns, tools, best practices and particularly, development methodologies and frameworks.

Famous waterfall. Agile Manifesto. XP and Scrum. No silver bullet, especially with usability.

Usability, HCI.

Startups as particular area of research. [8]

Usability and Agile is well researched and well documented.

Merging of those domains has little literature - hence the reason for case study. It's important because will provide guidelines for industry practitioners.

2 Introduction

2.1 Brief history of software development process

Nowadays, agile methods are without a doubt an IT industry standard. But many think of them as just a replacement for waterfall, while iterative and incremental development dates as back as to mid-1950s, with few well documented undertakings by IBM and NASA [6].

2.1.1 Waterfall

Waterfall became well-known after 1970 Winston Royce's article "Managing the Development of Large Software Systems". It promoted well-defined, strict process, consisting of gathering requirements, analysis, design, coding, testing and maintaining. Even though original article advised following those activities twice, waterfall was most known as it's single-pass version and became a standard for long years, especially in large-scale and enterprise projects. Craig Larman [6] argues that, as famous H.L.Mecken quote says "For every complex problem, there is a solution that is simple, neat and wrong.", waterfall gave illusory sense of "orderly, accountable, and measurable process, with simple, document-driven milestones", was easy to explain and recall (much easier than iterative and incremental development practices already present at that time) and widely promoted in literature.

2.1.2 Iterative and incremental development methodologies

Back in 1970s iterative and incremental development (IID) approach was already becoming recognized as more natural and fitting to manage projects than waterfall, notably IID incorporation by IBM Trident submarine system with above 1 million lines of code and NASA's space shuttle software. In 1976 "Software Metrics" Tom Glib proposed "evolutionary project management", and argued that system should be implemented in small steps, producing the appearance of stability in order to "have opportunity of receiving some feedback from the real world before throwing in all resources intended for a system".

1980s consisted of vast amount of criticism in literature towards unquestioned dominance of waterfall in industry [6], advocating IID approach, with notable 1985 Barry Boehm's publication "A Spiral Model of Software Development and Enhancement". It introduced risk assessment and project review in each iteration along with development phase using appropriate management model. During that time there were well documented significant project failures by US Department of Defense using document-driven waterfall approach, which resulted in adjusting formal standards (DoD-Std-2167A) to encourage usage of IDD.

In 1990s, software development industry was much more aware of IID practices and there were many books, articles and standards promoting IID. In second half of 1990s most currently known methodologies were created - Scrum, eXtreme Programming (XP), Rational Unified Process, Dynamic Systems Development Method and Feature Driven Development.

In 2001, a group of 17 methodologies experts, including all of above mentioned met in Utah to discuss common practices and the term “Agile” was coined.

2.1.3 Reigns of the Agile

Meeting in 2001 resulted in famous Agile Manifesto [1]:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Agile clearly promotes lightweight process and has been well received in industry and increasingly adopted, with recent 11th Annual State of Agile Report in 2017 [7] stating that over 94% of over 20,000 respondents claimed their organizations practiced agile. Among agile practices, Scrum is undeniably the most commonly chosen methodology (58% respondents) with Scrum/XP hybrid taking the second place (10%).

Since the creation of agile, the industry has also moved forward by leaps and bounds in technical areas. Automation is now common in software process - quality assurance, continuous integration and continuous delivery, internet as development environment, distribution means and execution infrastructure. Not to mention great progress in languages, tools, frameworks and architectural solutions. [3] The progress is unquestionable and although large IT projects were once notorious for spectacular failures (famous E.Yourdon’s “Death March” [12]), reported project success rates (meeting budget and time constraints and project goals) in various studies from went from barely 20-30% in 2000s [5] to 60-70% in 2017 [9].

One particular area, where statistics are not optimistic are startups, where “great majority of such companies fail within two years of their creation” [8] and “more than 90% of startups fail, due primarily to self-destruction rather than competition” [4].

During recent years of “agile methods reigns” there came natural realization that agile approaches are not silver bullets as they have limitations due to organizational and development environments in which they are incorporated and that they can be extended to address their limitations. Paradoxically, those extensions are often in the form of more “traditional” development process approaches [11].

Next section will cover state of the art of proposed agile extensions.

2.2 State of the art - Agile and Usability

Recent agile - related research has diverged into many branches. One of those tries to quantitatively answer to questions such as what particular practices are used the most in the industry (time boxing, planning meeting, daily discussions) and what domains are they mostly used in.[2] Summarize the one about Architecture.

2.3 Usability

Brief history to summarize. [10]

2.4 Startups as area of interest

3 Problem formulation and methodology

4 DNCR Documentation

5 Solution evaluation

6 Conclusion

References

[1] Kent Beck, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, et al. The agile manifesto, 2001.

[2] Philipp Diebold and Marc Dahlem. Agile practices in practice: a mapping study. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, page 30. ACM, 2014.

[3] Alfonso Fuggetta and Elisabetta Di Nitto. Software process. In *Proceedings of the on Future of Software Engineering*, pages 1–12. ACM, 2014.

[4] Carmine Giardino, Xiaofeng Wang, and Pekka Abrahamsson. Why early-stage software startups fail: a behavioral framework. In *International Conference of Software Business*, pages 27–41. Springer, 2014.

[5] Rupinder Kaur and Jyotsna Sengupta. Software process models and analysis on failure of software development projects. *arXiv preprint arXiv:1306.1068*, 2013.

[6] Craig Larman and Victor R Basili. Iterative and incremental developments. a brief history. *Computer*, 36(6):47–56, 2003.

[7] Version One. 11th annual state of agile development survey, 2017.

[8] Nicolò Paternoster, Carmine Giardino, Michael Unterkalmsteiner, Tony Gorschek, and Pekka Abrahamsson. Software development in startup companies: A systematic mapping study. *Information and Software Technology*, 56(10):1200–1218, 2014.

[9] PMI. Pmi's pulse of the profession 2017, 2017.

[10] Frank E Ritter, Gordon D Baxter, and Elizabeth F Churchill. User-centered systems design: a brief history. In *Foundations for designing user-centered systems*, pages 33–54. Springer, 2014.

[11] Dan Turk, Robert France, and Bernhard Rumpe. Assumptions underlying agile software development processes. *arXiv preprint arXiv:1409.6610*, 2014.

[12] Edward Yourdon. Death march: The complete software developers guide to surviving mission impossible projects. paul, db, 1997.