Do the benefits from early and frequent user research outweigh the potential cost on the return on investment for games?

COMP150 - Agile Development Practice

1703170

November 16, 2017

One of the major factors that contribute to a games success at launch is the ability to recognise and fix problems with the game before release. But does too much user research and playtesting negatively impact development? This paper outlines how user research is implemented using Agile development methods and how these methods affect development time and costs compared with other less rigorous methods in the games industry.

1 Introduction

User research is a vital part of a games development and allows a developer to gather important data about the game like metrics (see fig.1). But how much user research is appropriate for your game? Production teams that take an Agile approach [1] to game development often implement user research early on in the process and yield positive results when implemented well, while others that prefer plan-based approaches to game

development do the majority of their playtesting and user research towards the end of the development process. Often these can lead to issues at launch. The purpose of this paper is to discuss the positives and negatives of an Agile and iterative approach to game design with reference specifically to that topic of user research. The paper will compare and contrast these findings to the methodology they used in development [2].

Table of Player Deaths

	Min	Average	Median	Max
colombia-museum-break-in-roof	1	1.9	1	7
colombia-chase-fence	2	2.2	2	3
colombia-rooftops-tiles	2	2.8	3	4
syria-syria-turret1-outside	1	2.3	2	7
syria-syria-rpgesus-trapped	2	5.9	6.5	9
syria-syria-area2-start	1	2.9	2	8
syria-syria-area2-return	1	7.5	8	14
syria-syria-escape-hub-exit-mid	1	3.0	2.5	8
syria-syria-escape-bridge	1	3.1	2	8
yemen-temple-yem-temp-exit-combat-mid	1	2.2	1	8
grave-grave-01-freighter-section-2-exit	1	3.4	3.5	6
grave-grave-01-firstyard-start	1	3.7	3.5	9
grave-grave-01-firstyard-combat-mid-left	0	1.4	0	10
grave-grave-01-firstyard-combat-mid-right	0	5.3	6.5	11
grave-grave-01-firstyard-wreck-hatch	0	3.1	2	9
cruise-ship-cruise-container-fight-mid	2	5.0	4.5	8
cruise-ship-cruise-ballroom-fight-start	1	5.4	6.5	10
cruise-ship-cruise-ballroom-fight-mid	3	7.6	8	11
cruise-ship-cruise-chandelier-climb	1	2.8	2	9
airport-car-field-start	1	2.3	1.5	8
airport-car-field-mid	1	2.3	1.5	7
sandlantis-san-desert-battle-start	1	4.9	6	7
sandlantis-san-cistern-noria-tower-start	0	3.1	2.5	6

Figure 1 [3]

2 Postmortem Comparisons

In Agile, the early research data gathered from playtesting can be used in sprint reviews and can mitigate the need for a large crunch at the end of development as bugs and design flaws can be spotted and fixed much sooner [4]. More traditional methods may not catch these in this early stage. The longer that problems go unnoticed, the more work is put into problematic code, the more difficult and time consuming it becomes to fix later on [5]. This clearly advocates the idea that early, frequent playtesting is

beneficial to development time and costs.

In more traditional methods like waterfall, testing is usually planned to be done at the end of a project [6]. If playtesting is only used at the end of development for Quality Assurance, this can overwhelm a studio. [7]. Playtesting is about improving quality whereas QA is about making the game fully functional. Only allowing a few months to playtest a game before the final development stage is a risk to developers and investors. If the scope of the project has been underestimated then when it comes to the testing stage, a lack of funds or time to adequately playtest and do user research can easily become a reality [8]. This can mean a lot more issues at launch particularly for large AAA games [9] or the game not being released at all. With an Agile approach, the continuous iteration can ensure that a shippable build is always available and with continuous testing and feedback, each iteration is more likely to be the best the game can be at that time. This significantly lowers the risk for investors, always knowing that launch is viable [10]. This suggests that test-driven development in particular seems especially useful [11].

Day one playtesting of a game by the developers through an early prototype can also be useful to focus the team [12]. If you or other team members as developers are experiencing issues first-hand, it makes it difficult to ignore gameplay problems. This is very effective for small indie studio's who are making games with a smaller scope and constantly iterating over a basic build [13].

3 How Early and How Often?

Arguably, the most appropriate time to start playtesting is as soon as the developer has a question about the game [14]. This is likely to be very early on in the project and with Agile, and a flexible design document, user research about the game can be gathered in early iterations. With plan-driven development, this is a problem because the team has a large or rigid design document, and will be less likely to want to make changes before

the end of development.

How often a game gets playtested depends on how often content or features are integrated and when special events are held. Playtesting throughout gives the developer a much better grip on the players experience, saves time and energy wasted on ineffective material, and gets the developer to focus on the nuances of a game. [15].

4 The Community

Early Access on Steam is a useful way of gaining user research data [16]. Games that don't use Agile won't be able to go Early Access, as ideally this will happen well before the game is complete. This creates further funding for the game, and adds an early player base of testers who can provide feedback during development [5]. Methods such as waterfall would therefore be unable to benefit from EA.

There are risks however, such as if the initial release on EA isn't very appealing, perhaps because the gameplay mechanics are lacking. Normally this wouldn't be a huge problem, developers would be able to fix this in later iterations if playtesting was conducted within production. When the game has been released to the public in Early Access however, this can be an issue. The game will be judged by both players and press. People may forget about the game and therefore never see any of the improvements.

5 Testing and Team Communication

Testing early on in the development process can also be a boon to team communication as test-driven development promotes early communication between the different teams in a studio or company [17]. Communication between the Product Owner, Testers, Designers and Programmers is paramount to ensuring a successful game is made [18]. Whereas if a studio elects to take a more plan-based approach or even an Agile approach with strict and inelastic procedures they may suffer [19], [20], communications may break down and

team morale can be adversely affected [11]. This would hinder the development process overall and increase development times and costs.

6 Conclusion

It's clear that studios should use iteration [21], [22] but should user research be part of that iterative process? Arguably, yes. Although much of the evidence presented in this paper is anecdotal and stems from research of various developers postmortems of their games, it is nonetheless a valuable insight into how user research is used in the development process of games today. Although using Agile doesn't rule out issues in larger companies, it does appear to limit them [23], [24]. To summarise, the evidence clearly suggests that continuous playtesting and user research throughout the development process is very effective, especially in smaller teams. An argument can also be made that larger AAA companies should consider playtesting more often, even if they opt for a more waterfall approach to production.

References

[1] M. Fowler and J. Highsmith, "The agile manifesto," *Software Development*, vol. 9, no. 8, pp. 28–35, 2001.

[2] C. Politowski, L. Fontoura, F. Petrillo, and Y.-G. Guéhéneuc, "Are the old days gone? a survey on actual software engineering processes in video game industry," in *Games and Software Engineering (GAS), 2016 IEEE/ACM 5th International Workshop on*. IEEE, 2016, pp. 22–28.

[3] R. Lemarchand, "Attention, not immersion: Making your games better with psychology and playtesting, the uncharted way," 2012. [Online]. Available: www.gdcvault.com

[4] M. Fridley, "Postmortem: Kingdoms of amalur: Reckoning," 2013. [Online]. Available: https://www.gamasutra.com/view/feature/197269/postmortem_kingdoms_of_amalur_.php

[5] S. Johnson, "'offworld trading company': An rts without guns," 2017. [Online]. Available: www.gdcvault.com

[6] F. Ji and T. Sedano, "Comparing extreme programming and waterfall project results," in *Software Engineering Education and Training (CSEE&T), 2011 24th IEEE-CS Conference on*. IEEE, 2011, pp. 482–486.

[7] J. K. Lauri Hyvrinen, "Postmortem: Frozenbyte's trine," 2010. [Online]. Available: https://www.gamasutra.com/view/feature/134198/postmortem_frozenbytes_trine.php?page=4

[8] C. A. Alberto Moreno, "Postmortem: Crocodile entertainment's zack zero," 2012. [Online]. Available: https://www.gamasutra.com/view/feature/173453/postmortem_crocodile_.php?page=3

[9] J. Henry, "10 video games that were horribly broken at launch," 2015. [Online]. Available: https://gamerant.com/broken-games-launch-batman-halo-diablo-155/

[10] K. Ghane, "Quantitative planning and risk management of agile software development," in *Technology & Engineering Management Conference (TEMSCON), 2017 IEEE*. IEEE, 2017, pp. 109–112.

[11] J. J. Cunningham, "Costs of compliance: agile in an inelastic organization," in *Agile Conference, 2005. Proceedings*. IEEE, 2005, pp. 202–211.

[12] M. Yampolsky and W. Scacchi, "Learning game design and software engineering through a game prototyping experience: A case study," in *Proceedings of the 5th International Workshop on Games and Software Engineering*, ser.

GAS '16. New York, NY, USA: ACM, 2016, pp. 15–21. [Online]. Available: http://doi.acm.org.ezproxy.falmouth.ac.uk/10.1145/2896958.2896965

[13] C. Cleveland, "Postmortem: Unknown worlds entertainment's natural selection 2," 2013. [Online]. Available: https://www.gamasutra.com/view/feature/187299/postmortem_unknown_worlds_.php?page=2

[14] I. Livingston, "Where are the sharks? user research in the far cry production pipeline," 2014. [Online]. Available: www.gdcvault.com

[15] A. de Jongh, "Playtesting: Avoiding evil data," 2017. [Online]. Available: www.gdcvault.com

[16] M. B. Robbins, "Games in early access," 2015. [Online]. Available: www.gdcvault.com

[17] R. E. Gallardo-Valencia and S. E. Sim, "Continuous and collaborative validation: a field study of requirements knowledge in agile," in *Managing Requirements Knowledge (MARK), 2009 Second International Workshop on.* IEEE, 2009, pp. 65–74.

[18] R. McDaniel, "Communication and knowledge management strategies in video game design and development: A case study highlighting key organizational narratives," in *Professional Communication Conference (IPCC), 2015 IEEE International.* IEEE, 2015, pp. 1–8.

[19] J. L. Cooke, *Everything you want to know about Agile: how to get Agile results in a less-than-Agile organization.* IT Governance Ltd, 2012.

[20] B. Davis, *Agile practices for waterfall projects: Shifting Processes for Competitive Advantage.* J. Ross Publishing, 2012.

[21] A. O. OHagan and R. V. OConnor, "Towards an understanding of game software development processes: A case study," in *European Conference on Software Process Improvement*. Springer, 2015, pp. 3–16.

[22] R. Al-Azawi, A. Ayesh, and M. A. Obaidy, "Towards agent-based agile approach for game development methodology," in *Computer Applications and Information Systems (WCCAIS), 2014 World Congress on*. IEEE, 2014, pp. 1–6.

[23] D. F. Rico, H. H. Sayani, and S. Sone, "The business value of agile software methods," *Maximizing RICO with Just-In-Time Process and Documentation. Fort Lauderdale: Ross Publishing*, 2009.

[24] D. Batra, W. Xia, D. E. VanderMeer, and K. Dutta, "Balancing agile and structured development approaches to successfully manage large distributed software projects: A case study from the cruise line industry." *CAIS*, vol. 27, p. 21, 2010.