Why does play-testing and having an always playable build in games development work so well with the agile methodology?

1708082

16-11-2017

When people think of video games they will often think of fun and entertainment. And that makes logical sense, games are made to be fun and at least engaging to the point of entertainment [1]. But often in games development, developers will struggle with making games exciting and innovative while also most importantly, keeping them fun to play. Often if a game has several unique features and game play elements, its harder to test as well because the developers will have nothing to compare it to [2]. So the question arises; how do you incorporate making a game enjoyable into the development process? That boils down to the methodology that is utilised by the developers and how they decide to incorporate it. In this paper the stage of iterative development in agile methodology will be explored in the context of game development to investigate why it is effective in creating video games.

The agile methodology is known for being particularly flexible and effective in creation, for iterative development [3]. What's beneficial about agile development in the context of games development is that there is no need to have a requirements specification. Instead, in the case of the scrum agile methodology, the developers use a task board that highlights what the end consumer would want from the game. And in accordance to the scrum agile methodology, development takes place in sprints which can range from anywhere between a few days to a few weeks. What this allows for is a form of iterative development that is more a process of refinement than it is separate stages of development. This allows for the game to become playable at incredibly early stages of development as it is considered essential for there to be a constantly functioning, always playable build to work with. With the backbone of the game developed, the developers can employ a process of rapid design, integration and ideation in which they use the latest playable version of the game as a testing ground for enjoyability and testing. Over a short period of time, a development team can create something that at least resembles the creative vision behind the final game and can continue to add complexity to it in further iterations [4].

We can see in real world examples how iterative development/refinement has helped to shape games into successful and respected products. The development of the cult classic Natural Selection 2 started with a simple level that displayed the core game mechanics [5]. From there they progressed by using their constantly updating playable build to keep adding and updating the game. They would play the game and decide what needed to be added or tweaked next and used that as a stepping stool for further development. This sort of development is a part of the ideology and nature of the agile methodology. You can also see how iterative development helped to create a successful game in the form of the 2015 game Rocket League. In high concept, Rocket League was developed over the course of several years and released in the form of several different independent games. They would release a finished game, assess it, and then work on the next version of that game. They eventually released Rocket League and had huge commercial and critical success [6].

The idea of having an always playable build is a core tenant of the agile development methodology and is part of the reason it works so effectively with games development. It goes back the beginning of this paper where it was reaffirmed that games have to be fun, entertaining and compelling otherwise they are not commercialisable. and the best way to test this is by actually going and play-testing the game to assess it's ability to engage a player [7].

An interesting benefit to the agile methodology in this context is how it doesn't necessarily put pressure on certain members of the development team. For example an artist is given more flexibility over what aspects of the game they create assets for as they can see how they fit into the game via the playable build. It gives them more time to see if they're assets need tweaking or refining and gives the rest of the team longer to assess the look and feel of the game. Similarly a designer is able to actually implement and test their levels quicker when there is a playable build ready and available for them to use [8].

Not everything about the agile development methodology lends itself well though to this form of iterative game development. Due to the iterative nature of the method, development is often at times accelerated and progress can go much quicker than expected. This sounds good as a high concept, but often a lot of changes in game development can go unnotice and this can result in cascading dependencies [4]. A cascading dependency is where several changes are made in multiple places to achieve a single goal. This can have an negative affect on development as it can lead to there being issues with the engine or the code that need to be cleared up to allow for further development. It can also cause efficiency issues which may impact the techincal performance of the game as well. Another issue that can arise is that developing an aspect to a game for the first time can be particuarly challenging when there is already a running build of the game [9]. And this can lead to complexities as every new function of the game has to be developed with the pre-existing game in mind, which can cause issues with the code and the assets.

Even with these negatives, the sheer benefits that the agile scrum method has for games development is overwhelming. It's evident that by using an always playable build in the success of games released using this methodology and it's evident from developers who use the method that agile is still currently the best method for developing games.

References

[1] William Swartout, Michael van Lent, "Making of a game of system design", Communications of the ACM - A game experience in every application, Vol. 46, Issue 7, pg. 32-39, July 2003. ACM.

[2] Elina M.I. Ollila, Riku Suomela, and Jussi Holopainen, "Using Prototypes in Early Pervasive Game Development", Computers in Entertainment (CIE) - Theoretical and Practical Computer Applications in Entertainment, Vol. 6, Issue 2, Article No. 17, April/June 2008. ACM.

[3] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas, "The Agile Manifesto", Agile Manifesto, Principles, February 2001.

[4] Minh Quang Tran, Robert Biddle, "Collaboration in serious game development: a case study, Future Play '08 Proceedings of the 2008 Conference on Future Play: Research, Play, Share, pg. 49-56. ACM.

[5] Charlie Cleveland, "Postmortem: Unknown Worlds Entertainment's Natural Selection 2", Gamasutra, pg. 2, February 2013.

[6] Alanah Pearce, Corey Davis, "How Rocket League Beat The Odds", IGN Expert Mode Interview, Episode 2, September 2017.

[7] Patrick Stacey, Joe Nandhakumar, "Opening up to agile games development", Communications of the ACM - Surviving the data deluge, Vol. 51, Issue 12, pg. 143-146, December 2008. ACM.

[8] André Godoy, Ellen F. Barbosa, "Game-Scrum: An Approach to Agile Game Development", Instituto de Ciências Matemáticas e de Computação, pg. 292-295, November 2010.

[9] Juyun Cho, "Issues and Challenges of Agile Software Development with Scrum", Issues in Information Systems, Vol. 9, No.2, pg. 188-195, 2008.