Using Agile Methods in Software Product Development: A Case Study

Jayakanth Srinivasan, Kristina Lundqvist
Mälardalen University

Abstract

The mythos surrounding the use of agile methods emphasizes improved customer satisfaction, developer morale, and end-product quality. While the difficulty of adopting these methods is mentioned, it is often glossed over in the discussion. This paper presents an in-depth case study of agile methods adoption in a software product development firm. The choice of the firm as the unit of analysis enables the identification of organizational, social and technological challenges with respect to using agile methods. Using a mix of interviews, observation and archival data, the evolution of agile adoption within the firm is reconstructed. The data analysis highlights the importance of the four areas of requirements management, scrum implementation, organizational learning, and verification & validation activities.

1. Introduction

Agile methods have been demonstrated to enhance customer value, improve organizational morale, and provide step improvement in product quality [1]. Yet, the discussions surrounding the adoption of agile methods have often glossed over the difficulties associated with the fundamental organizational change that is required to make the transition successful. When the added complexity of product development is thrown into the mix, in addition to the technical aspects, the organization and social aspects of software development and sustainment also become more pronounced. In this paper, we present an in-depth case study of the agile adoption journey of a software product development firm, GameDevCo.

The research was motivated by the larger question of how organizations balance agility and discipline in software development. Unlike the work in Boehm and Turner’s book with the same title that focuses on providing organizations with an effective tool for understanding and managing the risks associated with the adoption of agile methods [2], this paper takes an inductive approach to articulating the challenges faced within a single firm in their agile journey. The data used in this paper was gathered through interviews, direct observation and archived project information. This rich mix of data gathering enables us to triangulate the inputs, and thereby increase the validity of the overarching findings and analysis.

We selected the people we interviewed to span the major business units within the firm, and provide adequate coverage of perspectives from senior leadership, software architects, project managers, product owners, developers, quality analysts and support personnel. Our primary data collection occurred in a one week period, wherein we interviewed 30 people using a mix of one-on-one and team interviews, with each interview ranging from sixty to ninety minutes. The interviews were transcribed, and the data was analyzed using a grounded theoretic approach to identify the key themes and develop recommendations.

In section 2, we present an overview of GameDevCo’s evolution, and highlight the critical events and some of the successes that they have had over the last eight years. Following that is a brief overview of the Scrum framework, to enable the discussion GameDevCo’s Agile Journey in section 4. We conclude with a discussion around the findings, and develop recommendations for organizations that adopt iterative methods for software product development.

2. GameDevCo

GameDevCo was born as a startup project that was designed and built on a university campus by developers who had a passion for the game of poker, and deep technological expertise in all three critical areas: server-side software, game engine, and client-side software. The game was initially designed to develop the game playing skills using ‘play’ money, but the success of the game overall, led to the creation of a product that would allow people to buy-into and play the actual games. The market success led to the creation of a small firm. The development work still happened in the university town, while business development and operations were carried out in the city. This birth phase of GameDevCo spanned two years between 2000 and 2002.

GameDevCo’s senior leadership team noticed that software was not being developed on-time, and on-cost. To add insult to injury, the constant bug-fixes and patches were being implemented in the ‘live environment’. As a result, they imposed a very structured development
process on the team based on the heavily plan-based development approach adopted at a major telecommunications company. In addition to imposing the structured development process, the operations team was made responsible for the deployment of the product. In this growth phase (between 2002 and 2004), they developed a B2B business model, in addition to the business-to-consumer business model to create complementary revenue streams. The increased popularity of poker in the world (partly fueled by the World Poker Tour), highlighted the need for continuously improving the product to meet increasing competitive pressures.

The structured development process really did not improve organization performance, and given that the development team was improving the product incrementally, GameDevCo’s senior leaders felt that the developers should be provided with the autonomy needed to make the decisions needed, while ensuring that some formal product data management was being carried out. In this maturation phase (between 2004 – 2006) they decided to adopt Scrum as the development methodology of choice. To gain competence in the process, they hired an outside consultant to train their entire workforce on using this new process. As they learnt about the strengths and weaknesses of their product, they started adding new features, and capabilities, resulting in even greater complexity. They realized however that they were having difficulty in evolving their product, hence they created a ‘Skunk works Team’ to architect a new product that would be as good as, if not better than the existing product. As they were institutionalizing the process, two critical events took place: The consultant that was mentoring their adoption of scrum, left the organization; and GameDevCo was acquired by a global conglomerate, whose core competencies was not online gaming.

When the data for this study was gathered in early 2008, GameDevCo was learning to deal with an additional layer management from a parent company that did not really understand their product/service portfolio. More importantly, since the parent company did not Scrum, they were imposing new requirements on the development team by adding on a ‘systems verification’ step. In the hope of bridging the growing gap between product management, operations, and development, all of GameDevCo’s operations were bought together to a single city location.

From a strategic perspective, GameDevCo can be understood as having four major value streams: the B2B value stream that focuses on delivering services to other businesses that are involved in online gaming; the B2C value stream that focuses on selling products and services to the end online customer; the IT&O value stream that manages the infrastructure necessary for operations, and also provides the verification and validation environments; and the GD value stream, that focuses on product development and product maintenance.

From an execution perspective, the product owners generate the high level requirements, which are then refined by the system architects and project managers, for implementation by the designers and developers. In order to meet their growth needs, GameDevCo uses a large percentage of contract staff who do not receive the same training as full-time employees. In addition to the core execution aspects, two other support elements, namely human resources and verification and validation are of interest in this study.

## 3. Understanding Scrum

The notion of using cross functional teams in software development can be traced back to a 1986 paper written by Takeuchi and Nonaka [3], on applying ideas from the game of rugby to product development. The concept was formalized for the software engineering community by Ken Schwaber in 1995 in [4]. Since its formalization, Scrum has become more widely adopted, and as Marchenko and Abrahamsson note is set to become the de-facto standard in industry [5]. Scrum is described by the Scrum Alliance as an agile software development framework with three roles, three ceremonies and three
artifacts. This can be visualized as three layers showing the ceremonies, the process and the roles, as shown in figure 3.

The three roles of Product Owner (PO), Scrum Master (SM), and the self-organized Scrum Team (ST), have distinct yet coupled responsibilities as summarized in Table 1. The product owner is responsible for the business value of the project. He or she creates a list of requirements (often called user stories) that are prioritized based on business value (deliver value to one or more stakeholders, often the customer). This list is further augmented with preliminary cost estimates developed by the scrum team, to create the product backlog. In preparation for the actual execution of the project, the PO presents the overarching roadmap and the release plan in the Sprint Planning Meeting. The ST in conjunction with the SM pulls features from product backlog, and refines them into the actionable tasks that populate the scrum backlog.

The actual execution of the project is carried out using time blocks called sprints that last between two and four weeks. The duration of the time block is dependent on the nature of the project, the people in the project, and the organization policies with respect to project management. During the sprint, the SM leads the team in the daily scrum meeting to create a shared understanding of the state of project. Every team member shares what they did in the previous day, what they plan to do today, and what challenges they faced. This shared situational awareness enables the team to surface and manage new dependencies, and equally importantly, enables the scrum master to identify and correct any real-time impediments to the team. One of the features of most agile teams is their use of a ‘project wall’ to enhance situational awareness – during the scrum meeting, the team members often physically move tasks (written on cue cards or post-it notes) between the sprint backlog, the ‘in-work’ section, and ‘completed’ sections of their project wall. The scrum master also has the responsibility of maintaining the burn-down chart – the amount of work that has actually been accomplished to date in the sprint. This chart is refreshed daily, and acts as an additional information radiator for the team to see progress.

At the end of the sprint, the scrum team demonstrates the developed software to the product owner, who assesses the effectiveness of the sprint and determines the necessary reprioritization of the product backlog to create the goal of the next sprint. The first half of the sprint review meeting also serves as an opportunity to review the big-picture effects of the product, including the market, technology and business impacts. The second half of the meeting is led by the scrum master, and is devoted to the retrospective. The retrospective provides the scrum team with the time needed to reflect on the process, identify improvement opportunities, and create strategies for closing the gap. The process starts over with the planning meeting for the next sprint. The simplicity of the framework masks the difficulty of making it actionable, as seen in the case of GameDevCo.

Table 1. Scrum Roles and Responsibilities

<table>
<thead>
<tr>
<th>Scrum Role</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Owner</td>
<td>Define product features and roadmap</td>
</tr>
<tr>
<td></td>
<td>Prioritize user stories based on expected business value</td>
</tr>
<tr>
<td></td>
<td>Manage the product backlog by adjusting and reprioritizing</td>
</tr>
<tr>
<td></td>
<td>Accept the product created by the self-organizing team</td>
</tr>
<tr>
<td>Scrum Master</td>
<td>Ensure that the team is functional and productive</td>
</tr>
<tr>
<td></td>
<td>Shield the team from external interferences</td>
</tr>
<tr>
<td></td>
<td>Coordinate meeting activities and carry out project management activities</td>
</tr>
<tr>
<td>Self Organizing Team</td>
<td>Does the actual work</td>
</tr>
<tr>
<td></td>
<td>Supports the effort estimation of the product backlog</td>
</tr>
<tr>
<td></td>
<td>Creates a shared sprint goal and selects an executable sprint backlog</td>
</tr>
<tr>
<td></td>
<td>Pulls work to create the sprint backlog</td>
</tr>
</tbody>
</table>

4. GameDevCo’s Agile Journey

GameDevCo’s agile journey raises interesting issues when viewed through a socio-technical lens. The roles-artifacts-ceremonies elements of scrum discussed in the previous section forms an effective foundation for discussing the current state of the process within GameDevCo. In keeping with the generally accepted
principle that an effective team size for a scrum is seven plus or minus two people, GameDevCo uses multiple scrum teams within the product development team. To support the coordination across the multiple scrum teams, they created the ‘scrum-of-scrums’ ceremony. Our observations of both the daily scrum meetings, and the scrum-of-scrums indicated adherence to widely adopted practices, and highlighted the atmosphere of trust within the teams. In the daily scrums, team members openly shared their challenges, discussed accomplishments, and plans for the current day. At the scrum-of-scrums meeting, we observed the team’s use of bug types to focus on critical problems. While we were not able to observe either the planning meeting or the sprint review meetings, the interviews revealed limitations in the implementations of both the ceremonies. In sprint planning meetings, the root cause of the problems are the maturity of requirements. The limitations of the sprint review meetings are associated with organizational learning.

The requirements issue also highlights the challenges with respect to maintaining the product backlog artifact. The sprint backlog is visually maintained on the project wall, which shows the current state of the sprint backlog, the work in progress, and the work completed. Since the scrum team updates and maintains this artifact daily, it has been extremely successful. One of the innovations in GameDevCo is the automatic generation of the burndown charts that are used both for team situational awareness, as well as upwards reporting by the scrum masters. Additionally GameDevCo makes extensive use of information radiators (in the form of screens showing build data, bug rates, overall project progress) within the organization – with mixed results. In the case of the new product development team, a large percentage of the software had not get been developed, as a result, most of project appeared red, either due to build issues or due to system verification and validation challenges, resulting in most of them ignoring the screens. From a roles perspective, the challenges were centered mostly around the scrum masters, and the product owners. track record of success has to some extent blunted the urgency of building up system V&V capability. The analysis of the interview data highlights two causes for this apathy: the lack of a system owner; and a lack of understanding of what V&V means in the product development context.

4.1. Requirements

As noted earlier, GameDevCo built the first version of their product through a strong foundation in the underlying technology (server-side, game engine, and client-side), and a deep passion for the game of poker. Since it started as a ‘proof-of-concept’, there is limited historical documentation on how the product was actually developed. The market success has resulted in continuous improvement of the product, carried out at a pace that has not always been conducive to documentation. As one senior architect noted:

‘It’s not that people do not want to generate documentation – there is a question of how much documentation is sufficient

Given that the product owners were only recently appointed (at the time of the study), the product backlog consisted of user stories at various levels of abstraction, which were often not consistent with either the current version of the running software or the product roadmap. When queried about this variance in the product backlog, one senior developer noted:

‘When we find a bug or get a new requirement set from the product manager, We get into a room, and hash it out together, agree upon the design. The challenge we have is that the decision stays on the whiteboard, and often we forget what we agreed upon’

The impact of the variation in requirements is also seen in terms of the amount of time that the development team spends in actually refining the requirement to the point where it is implementable. When queried, one senior team member said that they spent anywhere between 50 to 75 percent of their time in getting it to a point where they could work on it. Since the scrum teams are often not involved in the initial schedule estimations recorded in the product backlog, the downstream effect of ambiguous requirements is schedule overruns and poor system quality. The poor quality is primarily due to the multiple scrum teams being unaware of the design dependencies and decisions made by other teams. The expectation is that the various teams would update the requirements/user stories that have been refined/changed in the online tool. The universal consensus in both GameDevCo’s management and the product team is that the tool is inadequate. As more than one person put it:

‘The tool sucks!’

This raises an important issue of having the right tools to support the process. When talking to one of the senior leadership members about the tool, they said that it was a corporate mandate, and given the vast investment made in the tool already, they would just have to live with it. As one of them noted:

‘Many requirements are poor but not all of them are bad – it has now become an urban myth that the requirements are all bad’

The impact on the organization as a whole is that critical design decisions are often lost. While the
expectation is that these decisions are either logged on the team twiki, or updated in the product backlog, we found that in the data we were provided that neither was the decision log in the twiki updated on a regular basis, nor were the user stories and their dependencies updated in the requirements management tool.

4.2. Scrum Implementation

When discussing the implementation of scrum within GameDevCo with the teams, one of the contract personnel noted:

‘I am a contractor who has been here for a year, and I still don’t know what scrum is’

This reflects the challenge associated with the growing size of the organization. In order to meet their strategic objectives, GameDevCo has had to hire contractors to join the scrum teams. The expectation with these personnel has been that they hit the job running, and hence do not receive any training in the process. It is left to the teams to educate the contractors on the process. Given the intensity at which these teams work, the peer training also gets lost. Another issue that comes up arises from the graphic design work that is carried out as part of the UI development – where the role of scrum is still unclear.

From an organization standpoint, there is still significant variation in how scrum is implemented, and what the expectations are from the process. As one of the scrum masters noted:

‘We spend more time arguing about what the scrum books say about the process, rather than in trying to figure out what the process is trying to do for us’

The role of the scrum master’s is another area where there is significant variation within the organization, with team member stating that the job of the scrum master was to take care of the ‘external stuff’, and that it sometimes came down to filling out reports and getting coffee for the team. The scrum masters themselves see their role as being a key facilitator for the teams, and ensuring that they can to some degree shield their teams from external influences – the challenge being that most of the scrum masters were hired from outside GameDevCo, and did not understand the culture of the organization or the teams themselves. Furthermore, these scrum masters were often experience project managers with limited exposure to agile methods. This lack of experience impacts the organizational learning significantly. When asked about the effectiveness of scrum in the organization, one of the teams noted:

‘We are not doing scrum, we are sprinting’

4.3. Organizational Learning

The sprint review meeting is designed to support organizational learning from the product knowledge/awareness perspective, as well as the process perspective. Since the scrum masters each have individual philosophies of what scrum is, and how it should be implemented, the learning from the sprint review meetings has become almost non-existent. As one of the scrum teams noted:

‘Our retrospect starts with the scrum master saying, What didn’t work in this sprint? What can we improve in the process?... We look around and in 5 minutes we are done’

The focus on process improvement supports teams that are just beginning their scrum journey in both gaining a deeper understanding of the process, as well as in taking ownership of the process. As the team matures, process focused retrospectives produce diminishing returns. The emphasis on sprints, takes away from the overall learning at GameDevCo as well. As one of the senior developers noted:

‘We could have short cycles in the beginning as there was not that many dependencies between the teams – now we need time to integrate, and time to get the systems to work together – but we are compressing the sprint planning and sprint reviews instead’

4.4. Verification and Validation

One of the challenges that GameDevCo faces is in getting discipline into system verification and validation (V&V). As the head of V&V noted:

‘We only started thinking about system verification as a capability 1.5 years ago – I was brought in to bring some discipline into V&V, but it has been a tough road. We have young developers here who have always been successful, they don’t know when a project fails’

This track record of success has to some extent blunted the urgency of building up system V&V capability. The analysis of the interview data highlights two causes for this apathy: the lack of a system owner; and a lack of understanding of what V&V means in the product development context. The documentation-gap has been further exacerbated by the lack of a single ‘belly button’, namely a system owner, who defines/controls the system level requirements. As one of the quality analysts noted:

‘We don’t have a systems owner – so you don’t know if the system-level is actually correct. Since we don’t
have good requirements, we had to hire an external consultant to reverse engineer the system level requirements. The consultant was very good for the first three months, but other people found out, and we lost him.”

This problem is even more apparent in the context of new product development, wherein the high-level architecture is understood, but the requirements are being refined through the creation of rapid prototypes. The transition to scrum as the development methodology across the entire organization shifted the unit testing burden on to the developers. The scrum team built the test cases based on the scrum backlog, and aggregated the test cases for each of the scrum tasks to form the overall test suite, instead of deriving the system-level tests and acceptance tests based on the product backlog. Among the developers working on the product version that is running live that:

‘The system is working fine with the test suites we built, and we have automated the testing process to the point where we are able to run all of the test suites overnight’

In almost direct contrast to that view, is the perspective of members of the QA team:

‘Testing is not just automation and running tests every night’

What gets lost in the almost adversarial relationship between quality analysts and the developers, is the fact that they are working towards a common purpose. Currently members of the QA team are not part of the actual scrum team; instead, they are a common resource pool that all the scrum teams working on the product use collectively. This organization structure is effective if the system requirements are clear, and the QA team can effectively plan its own sprints to support the overarching product roadmap and expected release cycle. But as discussed earlier in this section, the system level requirements are not known for either of the two projects studied. From an infrastructure standpoint, the V&V team does not have a test environment that replicates the operational environment. As a result, they cannot determine if the system under test will actually perform as expected under live conditions. This is further reflected in the long system integration and verification cycle times, making V&V activities even more ‘unpopular’ with the scrum teams.

5. Discussion

Product development and sustainment is more analogous to a marathon, rather than a sprint. While it is possible to sprint a marathon, it requires tremendous discipline and stamina. From an organization perspective, GameDevCo recognized the need for increased team autonomy to support better product quality and enhance organizational innovation. As we have seen in the case study, there are four areas that GameDevCo has to focus on, namely: Requirements Management; Scrum Implementation; Organizational Learning; and Verification and Validation.

An agile process requires an agile tool – this challenge is present in spades. While the tool that is currently in use in GameDevCo works for managing archive quality requirements, it has not been able to meet the rapid development cycle time, as design refinements occur in parallel. The scrum framework can be effectively used in product development, but as with any process, the impact of what the process is doing for the organization is more important than the ideology of the process itself. That is not to say that ideology is not important – it is critical that the ideology be consistent across GameDevCo. The fragmentation of the implementation can be traced both to the loss of their agile coach, as well as their hiring of relatively inexperienced scrum masters. With a rapidly evolving product, and a mobile talent base, organizational learning becomes critical. The process does support learning, however the supporting infrastructure, and the implementation of the process itself does not support learning. Last, but not least, GameDevCo has to build verification and validation capabilities to support product evolution.

6. References