AGILE SOFTWARE DEVELOPMENT

Introduction

The following narrative paper illustrates the concept of Agile Software Development based on the development of a Polls App in a fictive German company.

ASD is a software development technique that shall ensure the timely delivery of well functioning software to the client by constantly testing and aligning the software prototype with the customer. It was brought to life by a group of American IT experts in 2001 when they summarized already existing ideas of a more agile approach in software development in a manifesto[1] (see "The Agile Manifesto" below).

Agile software development approaches incorporate various techniques such as pair programming, test driven software development, extreme programming and others. All these techniques share an iterative intention that aims to find the optimum solution for the client using various small trial and error processes.

The Agile Manifesto[1]

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

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

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

The 12 Agile Principles[2]

The core values found in the Agile Manifesto translate into 12 principles that characterize ASD approaches:

[1] Source: http://agilemanifesto.org/
[2] Source: http://agilemanifesto.org/principles.html

1. Customer satisfaction by rapid delivery of useful software
   -> Continuous adjustment of the product to customer requirements
2. Welcome changing requirements, even late in development
   -> Promotion of feedback culture
3. Working software is delivered frequently (weeks rather than months)
   -> Frequent and fast iterations
4. Working software is the principal measure of progress
   -> Design and perfect structure secondary
5. Sustainable development, able to maintain a constant pace
   -> The project is divided into several sprints that follow without gaps
6. Close, daily cooperation between business people and developers
   -> User and customer feedback needs to be accepted and implemented
7. Face-to-face conversation is the best form of communication (co-location)
   -> Co-location avoids misunderstandings and idle time
8. Projects are built around motivated individuals, who should be trusted
   -> Flat hierarchy structure, flexible role assignment, knowledge sharing
9. Continuous attention to technical excellence and good design
   -> Pair programming promotes "Get it right the first time"-principle
10. Simplicity—the art of maximizing the amount of work not done—is essential
    -> Reduction of unnecessary features, slim and fast processes
11. Self-organizing teams
    -> Flexible changes in team role assignments
12. Regular adaptation to changing circumstances
    -> Flexible software design and constant implementation of feedback

Main Story

Characters

Daniel: Chief HR Executive
Cliff: Daniel's Assistant
Brian: HR Analyst
Josh, **Audrey**, **Mary**, **Shawn**: Student Interns / Newly Hired Programmers
Steve: Brian's Contact within the IT Department. With FBar Corp. for 8 years now.
Jack, **Sarah**, + **2500 others**: Employees

Prequel

"I want it now," Daniel said, laying his hand flat on the conference room table. "Our internal communication needs to get better soon. The Top-100-Employers-Survey will go out to our employees in three months, and until then I want everybody in this firm to be able to participate in decision making. This will enable the management to take our people's opinions and convictions into account. I am confident that this process will lead to higher satisfaction and commitment among our employees. Brian, my assistant Cliff will send you the

specifications for our polls app this afternoon." Brian sighed. His boss had made up his mind, so there was no doubt what his occupation for the next three months would be. In his mind, he evaluated the options: Who could take on this task? His company, FBar Corp., did not have an endless stream of IT resources at their disposal, and this project idea came out of the blue. FBar Corp was a producer of high-class laser technology and had about 250 employees at its main production center in Koblenz.

In order to get a better picture of the available resources for this project, Brian called up Steve, a programmer that he'd known for several years and had a good personal relationship with. "Steve," he said, "Daniel has just given me an extra project to work on - high priority. What he basically wants is an app on our intranet that lets us evaluate our workers' opinions on all sorts of different corporate decisions. We need a running version in two-and-a-half months' time, at the end of August." "Gosh," Steve replied, "that's a tough one - I don't see how we could realize this project in only ten weeks. The entire department is busy with the latest ERP system update. But let me think about what we can do about this. How about a quick lunch tomorrow?" "Sounds great, I will have the requirements list from Cliff by then." "Ok, see you then."

The next morning, Brian printed out Cliff's email and read it briefly. As usual, Cliff had been eager to make his boss's wish come true, but the requirements list showed that Cliff had very little idea about how to implement this project. The email read as follows:

"Brian,

As Daniel discussed with you earlier, he expects the Polls App project to be completed by 31 August. Let me detail the requirements for the Polls App project for you:

1. The goal of the Polls App is to improve the employees possibilities for participation in the company's decision processes.
2. The App should allow each employee to participate in polls that have previously been set up by management (or their assistants, respectively)
3. To keep the poll simple and intuitive, Daniel would like to ask the employees three questions every three months. The App should allow us to implement a maximum of five answer options among which the poll respondent can then select. The first poll needs to go out to the employees in the first week of September, NO delays will be tolerated!
4. Daniel wants to see the results of the polls in a "C-level-compatible" format. For my needs, it suffices if you simply make the results from the polls Excel-downloadable so I can brush them up and convert them into management reports.
5. Since Daniel is a real iPhone-lover, he would absolutely LOVE to see the App be implemented on the mobile platform as well so that our employees can download and answer the polls even when they're not at work.
6. Implementation should focus on a smooth implementation into our intranet. Meanwhile, the stability of our intranet is paramount as we have several ongoing production batches that will not tolerate any IT-related delays.

7. The programmers can build the Polls App in any programming language that they like, but it is important that the App is stringently documented and adheres to our internal and externally given quality standards (see Guideline from the COO as of Feb 21).
8. Unfortunately, Daniel has not granted any significant budget for this project. He does allow you to charge 2 FTEs (Full-Time Employees) for the duration of the project though.

I am looking forward to hearing from you about the project progress! Let me know if you have any trouble.

Best,
Cliff"

What a mess, Brian thought. With this requirements catalogue and the operational restrictions, he could already foresee that the project would not finish before the Top-100-Employers survey is distributed. With that thought in mind, he entered the conference room, where Steve was already waiting.

"So, what do you propose for the project?" Steve wondered.

"I was thinking about taking one of your guys off the ERP project and staffing him on this Polls App."

"Well, you know how tight the schedule is. I don't think we can afford this."

"But Daniel wants us to give this project priority. We have no other choice, have we? I mean, there is a lot of stuff to be done: We need to double-check the project specifications with Cliff, kick off the technical design, set up a proper programming plan, modularize the code so that we can work on it simultaneously, and ultimately set up the testing environment, do the UAT and sign-off with Daniel - and all of that in only ten weeks!" Brian handed Steve the chart below, along with the printout of Cliff's email.

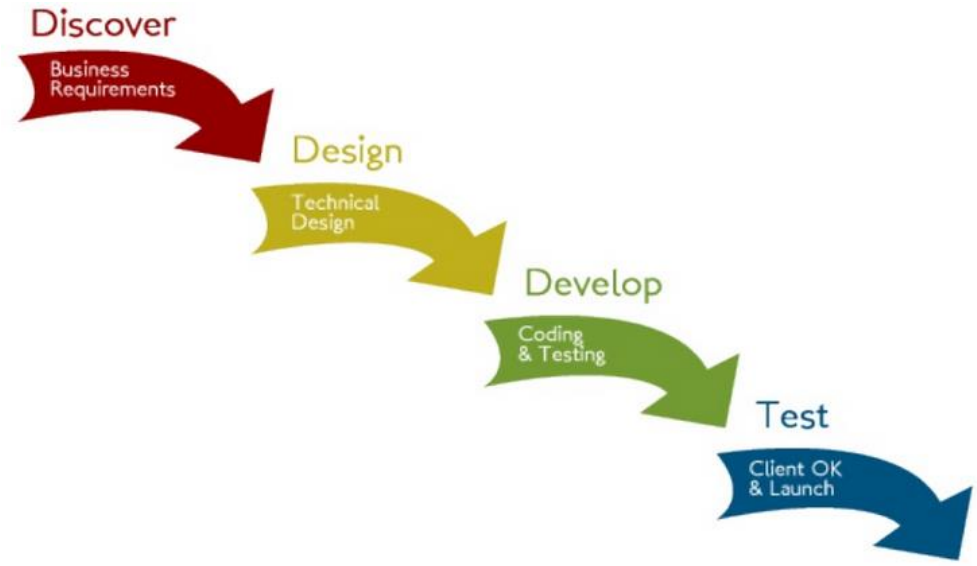

FIGURE 1: TRADITIONAL WATERFALL APPROACH[3]

[3] *Synerzip. "Building Enterprise Class Software". March 2013.* Available under http://de.slideshare.net/HemantElhence/agile-softwaredevelopmentmarch2013new

“Look, I think this project is plain unrealistic in the given timeframe,” Steve commented, “but perhaps we just need to change our approach. I have met a group of recent University of Koblenz-Landau graduates that introduced the concept of agile software development (ASD) to me. I’ve heard colleagues from other companies talk about it before. Frankly, I am convinced that agile software development could help us get this project done in the given timeframe and maybe pioneer a new IT R&D paradigm for our company. My suggestion is this: Since these students were quite interested in our company, let’s take them in as interns and have them work on the Polls App project. Since they are not at all familiar with our software development process, we need to employ a **leaner project management approach** to this project. Agile software development is faster and could help us get this App running. The students told me a lot about agile software development and their previous experience with it. In short, it is about making the project sponsors happy through **rapid software development**. While not uncoordinated, the agile software development paradigm **welcomes change as an opportunity to improve**. In order to achieve this, development is always aimed at the end goal of **bringing a running software to life**. Another important aspect is feedback: Through **frequent builds of the final software**, **customer feedback** is obtained frequently which is then used to further ‘evolve’ the software. While we measure the progress of the project in terms of to what extent the software is working, programming quality is upheld through **close collaboration among the team members** and a **regular exchange of ideas** among the programmers, the HR folks involved, and the employees that would use the App. In contrast to the quite rigid project management approach that we’ve used so far in this firm, agile software development gives the programming teams space to develop and **self-organize**, thereby enabling superior performance and creativity at the same time. And in order to make this project realistic in the given timeframe, we also need to care for **simplicity**: Everything that’s not necessary to get the App running will be put aside.”

“Wow, that’s radical,” Brian said. “Do you think the kids can get the job done?”

“Absolutely. They seemed to really love agile development, and they

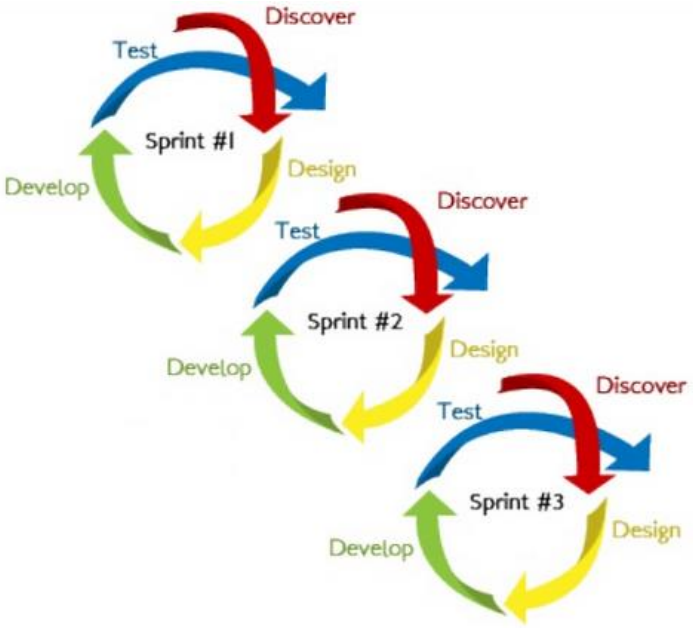

FIGURE 2: THE AGILE SOFTWARE DEVELOPMENT APPROACH[4]

[4] *Synerzip. “Building Enterprise Class Software”. March 2013.* Available under http://de.slideshare.net/HemantElhence/agile-softwaredevelopmentmarch2013new

would jump at the chance to prove themselves for FBar Corp. What do you think, can we get them on board as interns as a test and get them to implement agile software development philosophy across the firm if they succeed with the Polls App?"

"Sure, why not. We always need capable programmers, and if they bring a fresh thinking into the way we tackle projects, this is a really neat solution for us to get the Polls App running and strengthen our IT department."

Steve promised to take care of the project and get the students aboard. He and Brian agreed that getting Daniel's feedback was the key to a successful project, as they both knew too well that Daniel was prone to changing his mind frequently. Brian recalled that Daniel wanted to see how far things got in three weeks, and he also wanted to see an almost-final version in week 8. In addition to these two meetings, Brian agreed to have a weekly progress meeting with the team. With this plan in mind, Brian and Steve parted.

Project Kick-Off

Steve decided to organize a kick-off meeting with the student interns Josh, Audrey, Mary, and Shawn. He knew that some of them had already gained programming experience through internships and all of them were enthusiastic about ASD.

During the preparation for the kick-off meeting, a thought crossed Steve's mind. FBar Corp. had always had a very rigorous view on software development discipline. Thus, Steve doubted whether his young team was serious about ASD or viewed the concept as an excuse to just start coding and neglect proper planning and documentation. He decided to probe this point during the kick-off meeting: "Guys, I really liked what you told me about ASD as a software development philosophy that is all about an efficient relation to the business side. I guess that the days when we received a detailed requirements list with countless functions that the end users might never use are over, and I feel that we need a change in our software development culture. Nevertheless, software development in a corporate context might be different from what you are used to when coding for university projects. You will work in our intranet so system-critical bugs are absolutely inacceptable! And please don't forget: We are a team of five. **ASD is not about just starting to code and skipping all the planning and documentation**. The business side tells us what features they need most and we then think about a way to include these step by step into the application. Our goal then is to **come up with a running version of the app every two or three weeks in order to get feedback** from the guys who will use our program as part of their day to day work. This ensures that we can **correct any mistakes or misunderstandings right away before they become costly** - in terms of both time and money wasted. ASD is also about quick changes and flexibility, nevertheless solid documentation and visionary planning are important. Especially as you are interns we need to make sure that our in-house software developers can understand the code

and are able to implement changes later on should you guys decide not to stay with FBar Corp. From what I heard about ASD, the requirements for the quality of the software development team are much higher compared to traditional waterfall development strategies. So I really put a lot of trust in you and am sure that you will master this demanding project. And let me me also mention that we also need to make sure that we keep everything lean! ASD is not about fulfilling every dream of the business guys, it is also about **keeping the app slim and concentrate on the really important functions**. Ok?"
"Sure, Steve. We're really excited to help FBar Corp. switch to ASD. I believe it's the future of programming," Shawn ascertained.
"Well, I wouldn't go that far," Mary laughed, "but sure ASD has potential, and I can't wait to see ASD being implemented for such a big project."
"Get this project done and we will make you job offers that you cannot turn down." Steve grinned. "We need talented programmers at FBar Corp. that bring a fresh philosophy to our IT processes."

After some more casual talk, the kick-off meeting concluded in a joint lunch. Steve had intentionally avoided the topic of responsibilities within the team, as he wanted the team to **self-organize** and **assign responsibilities based on team members' individual capabilities** and the project requirements rather than based on a rigid hierarchy. But apparently, the students were at complete ease with that, and after lunch, the students had made up their minds as to whom would be doing which part of the project in the first three weeks. Audrey had already gained some experience in quality control through a student internship, so everyone agreed that she should take responsibility for this important matter. Mary was very business-savvy, so she would ensure that the project was well aligned with business requirements. Josh and Shawn had collaborated on a few university projects before and thus took over the project management and collaboration aspect of the project. This included choosing an appropriate working environment. Since FBar Corp. did not have a project management tool that supported agile development, they chose to use **Planbox**, a simple and intuitive agile project management tool.

Audrey mentioned the concept of **pair programming**. She explained that in ASD the risk of producing bugs is especially high. Hence she insisted on especially **rigorous testing and quality assurance**. In this context it made sense to team up in pairs of programmers so that **one person wrote the code while the college assured that the code was bug free and fitted the problem** it was intended to solve. In case any problems surface, the two programmers could discuss a solution and solve the issues immediately. She also explained the role of testing to her team colleges: "In ASD testing is a key factor, especially when using

techniques such as **extreme programming** that carry an extraordinary risk of producing bugs or running into the wrong direction. Therefore the testing has to be twofold and needs to be as agile as the software development process: On the one hand we need **continuous feedback from users** to show any mismatch between our application and the user requirement. In addition we need to limit the risk of bugs. **Pair programming** is a step in the right direction, but in addition we need to employ **automated unit tests** that assure the consistency of the developed code."

Week 1: Planning, UML Modelling and Coding

In discussing the timetable of the project, the team agreed to come up with a **limited but fully functional version** of the application at the end of the third week. This would allow for a first **public beta test** within the marketing department to gain some insights into which features are well accepted by the users early in the project. Therefore the team agreed on starting the planning phase already in the very first week by translating Daniel's requirements list into a UML diagram. As the App was set out to run on the intranet of the company, the student team pushed on using Python as programming language, although Steve didn't have a lot of experience with it. The students' telling argument was that **Phython is especially well suited for ASD purposes** as it combines the power of compiled languages with the simplicity of scripting languages [3].

In order to be able to fulfill the tight schedule of the project, the team **prioritized the different requested features** and thought about an order to start programming. According to the ASD philosophy, the key goal was to **come up with a running version as soon as possible in order to obtain feedback**, so the scope of the initial (week 3) prototype of the App was reduced to a primitive App that could run three questions and four answer options for each. It was clear that there would be some reductions in the visual features and that it might not be possible to analyze the data the App would return, but the App would already show the desired basic functionality. Josh insisted that it was crucial to have at least three weekly meetings with the HR Analyst Brian in the meantime in order to **ensure that the team headed in the right direction and to avoid any potential misunderstandings** already early in the process. These meetings would be about **aligning the features and user interface with the business needs**. Since Brian had no programming background whatsoever, the team aimed to present **mini-builds** of the software so Brian could actually see the product developing and hence would result in **earlier result visibility, risk mitigation and flexibility to accommodate change**.

Within the first two days, the team drafted a **UML model of the final (envisioned) Polls App** and completed a realistic assessment of what was possible to incorporate into the first prototype. With the outline of the project now clear and the **Planbox workspace** set up,

the team started their **first programming sprint**. They constantly tried to **limit all non-operational activities such as documentation** to the minimum amount required to make the project comprehensible to outsiders. This meant that the code was commented while it was written, but the formal documentation of functions and builds was deferred until a after the first functional prototype would have been built.

Mary also quickly called Cliff, Daniel's assistant, in order to check if it was ok to start a first public beta of the Polls App in week three in order to obtain broader feedback on the appropriateness of the functionality. This would allow the team to evaluate whether the App was going into the right direction so that they could **quickly calibrate their efforts**. Cliff responded by saying that Daniel thought this to be an unusual approach, but liked the idea of a running prototype at the end of week three and fast user feedback cycles. He advised to use a few innocuous test questions and make the Polls App accessible only to a limited group of workers, such as the HR department members.

Week 2: Realignment of programming and business needs

In week two the software developing team worked long hours in order to come up with a first running version of the App. Since Daniel had put emphasis on the point that he wanted to see a mobile solution of the App and because Audrey and Josh had already had developed a mobile App before, the team decided to develop a very simple mobile version of the Polls App along with the regular intranet version. The desired mobile device compatibility proved to be a difficult thing to tackle as the smartphone app needed to run in an entirely different environment as the intranet version. In addition it was not clear whether the smartphone app could be safely connected to the company's oracle database. Facing these substantial hurdles, the team decided to **check back with Brian to make sure that they were heading into the right direction**. A clarifying discussion (including a call with Daniel) made clear that Daniel's assistant Cliff had misunderstood his boss's intentions: Daniel wanted to make sure that the Polls App on the intranet could adjust to different browsers and screen resolutions but did not request a specifically developed smartphone app. The team, glad to have **realized their misalignment so early**, tabled the mobile version of the Polls App and redoubled their efforts on the intranet version.

Week 3: Public beta release with limited functionality

After another week of intense programming and coordination, the **first full prototype** of the Polls App was put on the intranet on Friday morning of week three. This version did already provide all basic functions including a mobile-browser-compatible view but lacked some of the fine tuning such as a well-designed user interface and the data analysis functions. The Polls App was initially only made available to the HR department. Questions asked were

trivial, such as “Which choice of coffee from our coffee machine do you prefer?” The HR employees were then asked to respond to the questions, but also provide some additional feedback regarding the usability of the App and any technical difficulties they might have faced. Brian collected and evaluated the answers from his colleagues and discussed the results with the student team.

The feedback showed that hardly anyone used the mobile browser as it seemed more convenient to simply answer the poll on the intranet which was open on all terminals anyways. As a result the ASD team decided (after checking back with Daniel) that the mobile feature should be dropped. In addition, Daniel told the team about a chat he had with the head of the marketing division who heard from the beta and liked the idea of being able to run polls in his own department. This feature was originally not intended as the limitation on the HR department was just due to beta testing convenience, but Daniel liked the idea and hence the team agreed on adding this functionality. Therefore division managers should have the functionality to generate polls for their own department using a user friendly WYSIWYG interface.

This showed that **testing the software during several betas and adjusting the software to the obtained feedback is one key success factor for ASD**. In addition it makes sense to **drop any unnecessary features** early in the process in order to **avoid spending time and resources on functionality that does not add operational value**.

Week 4: Adding charting features, discontinuation of mobile

In Week 4 the developers thus quickly executed Daniel’s decision and Audrey and Shawn, who had programmed the mobile interfaces, reversed their work towards mobile integration. At the same time, Mary and Josh met with Brian to talk about the best ways of illustrating survey results to the poll admin. Of course, they were trusted with this task, as team member and task distribution consistency is key for ASD. Also their **gained experiences and partly-undocumented knowledge** of how well-designed interfaces look enabled them to **quickly finish the sprint within just two days**. In short, the goal was to develop simple yet effective methods and designs to make sure that the scarce time of the polls admins would not be spent on unwanted features. To do so, the three worked together with sample data and the two developers quickly programmed a basic bar chart, **tested its effectiveness and understandability with Brian** and rejected the method for its low result visibility. So after another two day programming sprint they tried the next chart type: pie-charts. All three of them found this to be the most effective illustration method and after testing various corporate designs, all of them were confident to have chosen the best yet simplest way of illustrating the poll results. **This iterative process is typical for ASD.** At the end of the week, the developers and Brian met again, and the team was able to demonstrate automatic result analysis of past

polls as well as a test poll they quickly tried on Friday with the HR Department. When showing the results to fellow colleagues of Brian, the design obtained positive feedback.

Week 5: Adding department wide poll functionality

In Week 5 the project team tackled the task of enabling department-only polls. For this, the four programmers quickly met and organized themselves in two different groups to a) select the wanted poll participants and b) to prohibit outside access. They touched back with Brian to ensure they understood the desired functionality well, but could not reach him at his desk, as he was out-of-town for a recruiting event until Thursday. After sending him an email, his answer specifically stated that he wanted each poll admin, mostly department heads, to be able to individually ask their employees some questions, but prevent all other employees from answering or seeing the poll. Due to the lack of communication the developers misunderstood Brian's somewhat blurry requirements and developed the tool in a manner that left the admin searching for the aspired poll participants within the entire company by name and excluding everybody else. Brian, however, wanted it to always be department-wide (or any other company body & organization), without the possibility of individually including/excluding anybody, as he feared legal issues, e.g. excluded (or forgotten) employees suing the firm for not letting them participate in relevant polls. However, the misunderstanding only surfaced on Friday when the team presented the result and enabled Brian to start a poll with his best colleague buddies. Quickly the team met again and Brian told them that this version could not be accepted by the company and that they would need to find other ways of implementing company subgroup-wide polls. He promised to help them manage the complexity of the hundreds of subgroups within the organization next week.

Week 6: Problem Fixing

On Monday morning of the next week the developers hence met for a crisis meeting and evaluated the situation. The principles of **no code ownership** and courage should allow the programmers to reuse findings from past projects and other developers to fix the problem in an efficient manner. Using alternative access rights solutions proved difficult, however, since the basic access rights module for the Polls App that had been developed in week 5 was flawed and the access rights issue was very project-specific. The team found it hard to use any of their work from the week before and **decided to start the sprint again with the version that everyone had agreed on at the end of week 4**. Brian decided to ensure progress by actually **co-locating** his work station next to the students' for the duration of this sprint, as the subgroup categorization would require a lot of input from him. In doing so they significantly increased their conversations on how to code the subgroup polls, which led them to finding and **stopping another crucial error before testing**. At the end of the week they successfully launched two

tests in the marketing department and discussed the latest access rights solution with Brian, who liked it.

Week 7: Second beta test

After having had completed two small daily tests the week before, the team was convinced that the current version of the Polls App was ready for another **beta test**. This is why week seven started with launching the second beta test on Monday afternoon. This time it was conducted in three departments at once, namely Marketing, Finance and again HR. The version used in the test did provide all basic functions and a well-designed user interface. On top of that, it included a data analysis function and the revised implementation of subgroups. **By analyzing findings of the second beta test run, small bugs were recognized** by the development team. Since a C-Level meeting was scheduled for the next week during which working software was planned to be presented, the team used the remainder of the week for debugging and updating the App.

Week 8: Obtaining the C-level OK and adjustment of user rights

As already mentioned, week eight's central event was the C-Level presentation where the current software version was presented to Daniel in order to get his feedback, and hopefully, his acceptance - all according to the agile principle of **frequent delivery of working software**. Although week eight was a late stage in the development timeframe set by Brian, the development team was open for feedback from Daniel and welcomed possibly **changing requirements**. In good foresight, they had used available time before the meeting to make the Polls App **code more flexible** so that they could save time in case any adjustments were required. And indeed Daniel did have some concerns with the updated version of the Polls App.

He feared abuse of it since there was no limit in the amount of polls each department was allowed to run during a given timeframe which, according to him, could have led to irrelevant, not business related polls being started. He wanted to restrict the amount of polls per department and month. Due to that, the development team had to change the app again. They had to develop the app further by including the feedback they had received from management. Brian promised to work together with the team till the end of the project and therefore kept his workstation next to the programmers'. But team spirits were high, as Daniel had generally liked the project outcome very much.

The team spent endless effort to cope with Daniel and his ever changing demands. However, **Brian's continued presence stimulated the young team to high performance** so that at the end of the week, the project team had finalized the Polls App, including an update of user rights with a restriction of two polls per department and month as desired by Daniel.

Week 9: Final testing

After some sleepless hours in week eight and a weekend full of coding, the long-awaited **final test phase** started on the Monday morning of week nine. The test ran till Wednesday afternoon in half of all departments. There were no challenging bugs and the project team received great feedback from various departments that were more than happy to finally have a platform to influence decisions made by management and therewith shape their workplace. Nevertheless, the development team used the time till Friday to optimize the App and make it ready for its launch next week. On top of that, **Daniel was provided with the results of the final test and the optimized version of the Polls App to give his OK**. He was very pleased with the final product and accepted the firm wide release of it in the following week. **Due to the sorrow testing and frequent discussions this final meeting did not result in any surprise** and as the risk of delivering an un-useful piece of software was limited.

Week 10: Final rollout

This was the week the whole project was launched successfully. The development team as well as Daniel kicked off the release of the Polls App and did several town hall meetings to explain employees the rationale behind introducing the Polls App and the advantages it produced for them. Furthermore, they explained possibilities and mentioned constraints when using it. The general reception among the employees was outspokenly positive, and Daniel's secret objective of lifting the employee morale enough to finally get into the Top-100-Employers list came true when FBar Corp. was voted the 87th best company to work for in Germany.

All in all, Daniel as well as the whole project team were satisfied with their final product and celebrated the introduction of it with a joint team dinner that Brian had organized. Agile software development, that was the unspoken consensus that evening, would become a central aspect of the IT department's project management approach, and Audrey, Mary, Josh, and Shawn would be an integral part of this effort.

Comparison of ASD and Waterfalls philosophies in key situations

Situation	Week	ASD Approach	Waterfall Approach
Requirements List	Pre-Kick-Off	• Only a preliminary idea of the project is required	• Holistic representation of the expected outcome required
Kick-Off Meeting	Kick-Off	• Self-Organization leads to motivation and optimal resource use	• Strict Separation of tasks leads to individual focus and expert knowledge that needs to be managed and shared

Timetable of the project	Week 1	• Regular sprints lead to tight time management • Deviations of expected results surface early	• Very Strict, time management, everything must be planned ahead, low flexibility to incorporate changing requirements
Problems faced while Mobile Device Programming	Week 2	• Close feed-back loops and close business interactions allow quick reactions	• Misunderstanding not have surfaced until end of project -> Unhappy Customer
User Acceptance and Functionality Test (Beta Test)	Week 3+7	• Fast prototype with actual users that provide supportive feedback	• Full Running Version Tests at end of project with detailed thus complex questionnaires • Problems are detected when changes are already costly
Implementation of Result Illustration	Week 4	• Trial & Error, working together with actual users • Prevents overspending	• Proposal of programmers at end of project, if rejected: new programmer proposal
Misunderstanding about Subgroup Restrictions	Week 5-6	• Easier to change • Early recognition due to close feedback loops • re-start from previous working versions	• Would not have surfaced until the very end, lengthy revisits necessary
Late-Stage Changes	Week 8	• Changes are allowed and welcomed rather than having to re-do more work later on	• Given the rather inflexible set-up, small changes in late stages might lead to necessary high costs
Final Testing	Week 9	• Few changes necessary as every part of the software was already audited before	• Huge problems can surface, making the aspired end of project date impossible
Presentations	Week 10	• Smooth transition as product "poll" is already well-known	• Big surprise effects • Implementation problems may arise

ASD Principles in the different Stages of the Story

		Time Period									
		Start Phase	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10
Principles	Principle 1		x		x							
	Principle 2	x								x		
	Principle 3	x	x	x	x	x						
	Principle 4		x						x	x	x	
	Principle 5	x										
	Principle 6	x	x	x		x			x		x	
	Principle 7	x				x		x		x		
	Principle 8			x	x							
	Principle 9					x						
	Principle 10	x		x	x							
	Principle 11	x										
	Principle 12	x										

Sources Used

1. http://www.ambysoft.com/essays/agileManifesto.html
2. http://de.slideshare.net/HemantElhence/agile-softwaredevelopmentmarch2013new
3. http://dl.acm.org/citation.cfm?id=1177267
4. http://agilemanifesto.org/
5. http://geekswithblogs.net/robz/archive/2007/12/28/real-life-examples-of-agile-development.aspx
6. http://www.allaboutagile.com/agile-development-cycle/
7. http://www.codeproject.com/Articles/604417/Agile-software-development-methodologies-and-how-t
8. http://www14.in.tum.de/konferenzen/Jass06/courses/3/presentations/AgileSoftwareDevelopment.pdf
9. http://en.wikipedia.org/wiki/Agile_software_development
10. http://agilemanifesto.org/principles.html