A LESSON IN CONFLICT MITIGATION: INTEGRATING DIVERGENT DESIGN PHILOSOPHIES

Katelyn R. Brinker, Rebecca C. Marcolina
Missouri University of Science and Technology, Rolla, Mo, USA

Advised By: Dr. Kurt Kosbar, Dr. Melanie Mormile
Missouri University of Science and Technology, Rolla, Mo, USA

ABSTRACT

The Mars Rover Design Team is dedicated to building a next generation rover that will one day assist astronauts in the field. The complexity of such a project increases when the two conflicting design philosophies of agile software development and traditional waterfall development must work in tandem in order to design and construct a rover within a year. Agile software development promotes the flexible, test-driven production of coinciding design aspects, while the waterfall design philosophy relies on thorough planning and rigid, sequential design schedules. The project managers of the team work to balance these opposing philosophies by fostering individual interests, allowing team members to select their own focus areas within a wide variety of mission critical tasks. This practice accelerates the design and construction of the rover and in turn creates the momentum needed to achieve a common goal while consolidating both agile software and traditional waterfall development.

INTRODUCTION

When managing a design team, taking the risk of implementing two conflicting design philosophies is an unconventional yet, in our experience, effective approach to project management. The Mars Rover Design Team (MRDT) of Missouri University of Science and Technology was created four years ago in order to compete against other schools in the University Rover Challenge (URC), an annual international competition in which schools design and construct rovers (Fig. 1) to execute a series of tasks similar to those critical to a future manned mission on Mars [1].

Figure 1: MRDT's 2016 Competition Rover, Zenith

The team's success at competition is enhanced by the use of effective project management throughout the rover's development, but it is primarily a product of every team member's cooperation and dedication to the design team. It is not always easy, though, to achieve cooperation when the team implements two divergent philosophies - traditional development and agile development - into the design process. The project managers of the team work to balance these opposing philosophies by fostering individual interests; thus, team members are encouraged to select their own projects within a wide variety of mission critical tasks. This practice accelerates the design and construction of the rover and in turn creates the momentum needed to achieve a common goal while consolidating both design philosophies.

THE DESIGN PHILOSOPHIES

While traditional waterfall development and agile development contradict one another, they are both effective models of project management. For this reason, MRDT works to utilize both approaches, in spite of the fact that the team faces conflicts as a result of this dual implementation. The traditional style of project management is a form of management that relies on sequential design and construction cycles. First officially proposed in the 1970s, the waterfall methodology can be broken down into the following basic phases: research, design, implementation and testing, verification and review, and maintenance (Fig. 2) [2]. These phases encompass smaller, phase-specific development stages, and once a particular phase is completed, it is typically not revisited. A linear, top-down methodology such as waterfall development has been found to break a project's life cycle into manageable pieces more effectively than some

other management styles, especially where large projects are concerned [3, 4]. All design constraints and requirements are gathered from the customer and researched thoroughly before designs are finalized so as to minimize design errors and miscommunications before construction begins. The critical aspect of this design philosophy is to accompany each phase with continuous, thorough documentation and design analysis. In doing so, progress can be tracked easily with clearly defined milestones, and most problems are uncovered and solved before entering the testing phase [3]. However, some aspects of a project, such as its software or electrical hardware, can be hindered by this rigid development structure. By testing near the end of the project's life cycle, it is possible that some design or implementation errors will go unseen, later causing elevated costs or schedule delays.

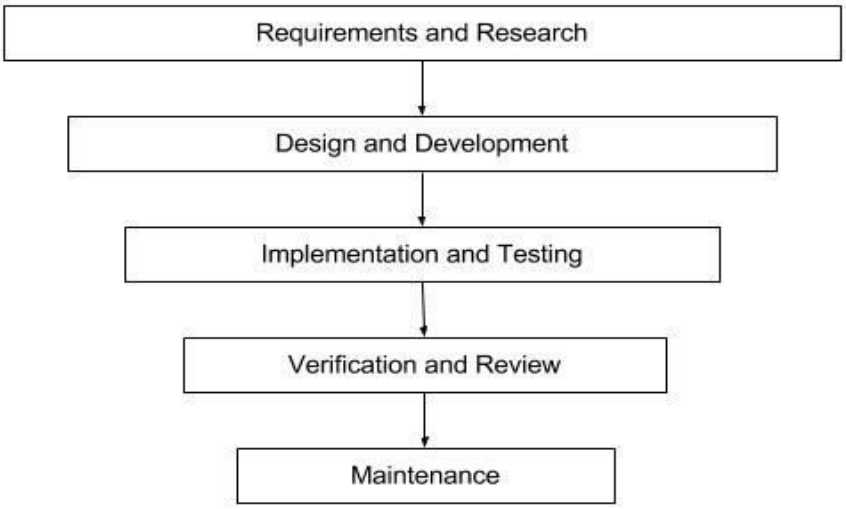

Figure 2: Breakdown of Traditional Waterfall Methodology

As a result, a new style of project management was introduced in the early 2000s to change these problematic aspects of traditional methods [5]. The agile methodology is a flexible, iterative approach to management. Via agile, cross-functional teams design and test coinciding aspects of a project throughout its development cycle while regularly submitting their results for customer review and feedback. Composed of developers, designers, planners, and testers, these different cross-functional teams use various iterations, or short phases of development, to develop individual aspects of a project [6]. Instead of an intensive initial research and design phase, iterations rely on small bursts of productivity followed by recurrent test phases, as detailed in Figure 3. Individual aspects of the project are later integrated into the rest of the system, tested, and subjected to feedback from senior analysts, customers, and business representatives [7]. If there are no further changes necessary, the finished aspect is then incorporated into the rest of the project, and the team is given a new task. If additional changes are suggested, then the team begins a new iteration cycle to redesign their component. Iterations have loose schedules, as they are organized in priority order based on values determined by the developer or customer [8]. This flexibility allows customer requirements and expectations to change throughout the development process, and while it can be difficult to track and document uniformly, progress is usually rapid when employing this methodology. Both methodologies can be advantageous to project development, but blending these conflicting design philosophies can create many complications.

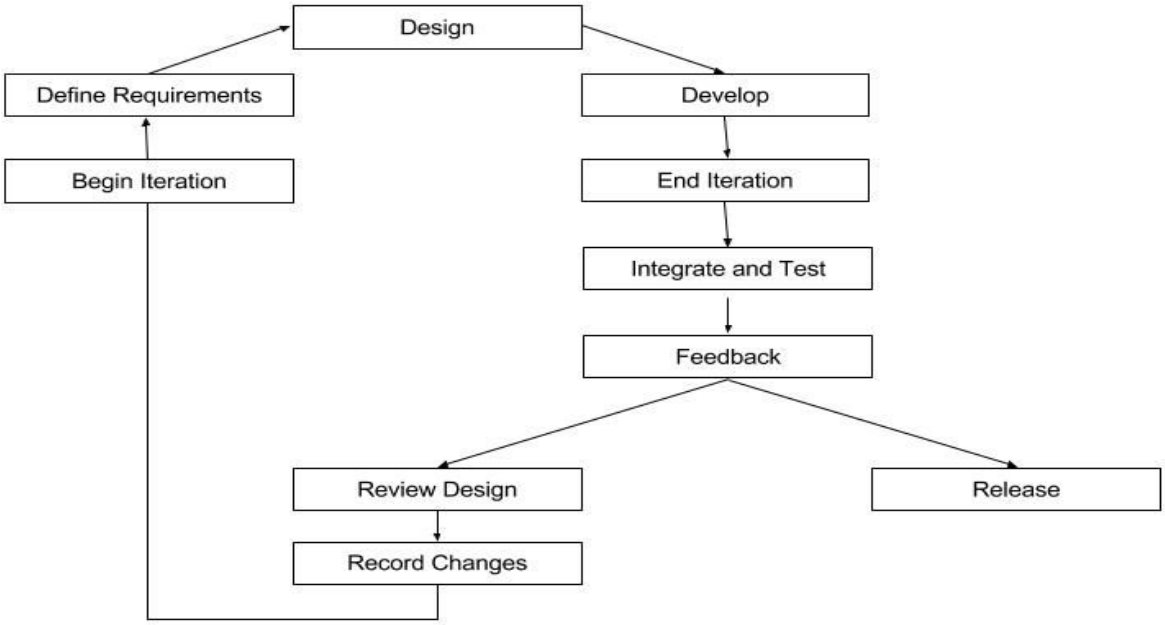

Figure 3: Breakdown of Agile Methodology

MRDT AND PROJECT MANAGEMENT

To understand which problems the team faced when these two methodologies clash, one must first understand the team's structure. The organization of MRDT is similar to that of a small engineering firm: a team elected, four-member executive board oversees the team's four branches - Executive, Administrative, Financial, and Technical - that break down further into smaller divisions as shown in Figure 4.

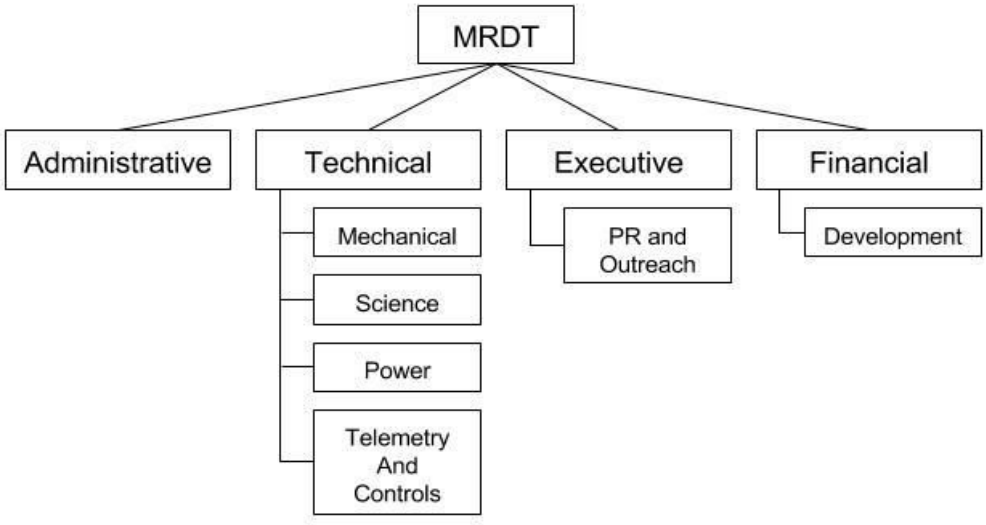

Figure 4: Overarching Team Structure

However, the challenges to project management brought about by mixing traditional and agile development originate solely in the Technical branch of the team, which is examined in Figure 5. Composed of four different sub-teams - Mechanical, Telemetry and Controls, Power, and Science - the Technical branch is responsible for the complete design and construction of the rover. Each sub-team also breaks down further into project-area based squads.

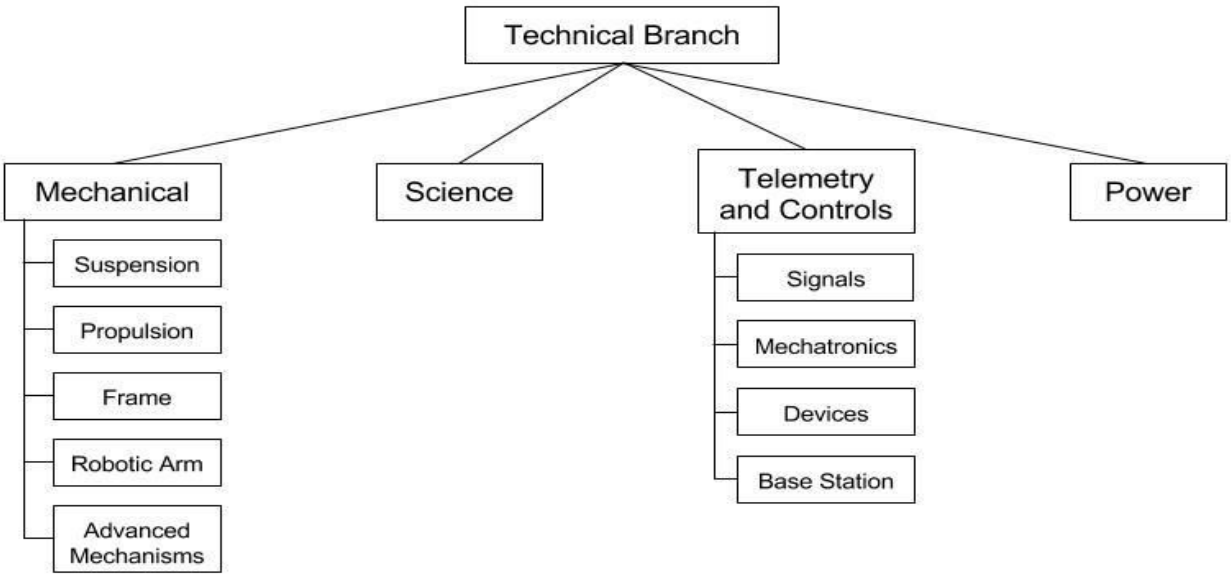

Figure 5: Technical Branch Breakdown

The Mechanical sub-team primarily utilizes the traditional waterfall methodology. This rigid, sequential approach is a sensible choice for a construction-based project, as strict schedules help the team meet URC-mandated deadlines for initial designs, models, and prototypes. The long research period of the waterfall approach guides the Mechanical division to stay within budget, and intensive system analyses help prevent major failures when testing parts. Attempting a linear, one-shot approach to software development created unnecessary delays for many Telemetry squads, as the requirements for a rover's software systems are often incomplete or ambiguous [4]. An iterative development methodology such as agile enables system changes to be made easily as the project progresses. When all critical systems are functional, squad members can then work to implement features that, while not mission critical, benefit the team's competition performance. The Mechanical and Technical sub-teams each implement the design approach they feel to be the best fit for their particular project, and while this practice is efficient within the two individual sub-teams, it often causes problems that spill over into the rest of the team.

IMPLEMENTATION AND CONFLICTS

In choosing to implement both traditional and agile design philosophies, MRDT members had to recognize that neither philosophy would exist in its pure state and thus, learn to compromise accordingly. For example, as a student design team, MRDT does not have a specified customer.

Instead, the team views its competition as a customer, basing design requirements and research off of current URC rules and observations from previous years. Given the fact that these specifications are often fairly vague, and that many systems are interdisciplinary in nature, the Technical branch's network of sub-teams and squads also regard one another as customers. To create most of the rover's components, squads from different sub-teams are required to collaborate on designs. This is where most conflicts arise, as each sub-team takes a different approach to the design and development phases.

In the team's early years, MRDT tried to exclusively use the traditional method. This was a major benefit for the Mechanical sub-team, as the rover's mechanical systems were fully designed and developed in time for two or more rounds of tests and refinements. On the other hand, this practice led to a lot of complications for the Telemetry and Controls sub-team. The sub-team's design phase lasted through the first four months of the project's life cycle, meaning that work on the systems did not start until January. Yet little to no progress was made when work began, as poor designs caused the core systems to be more complex than originally anticipated. Telemetry squads had to try and salvage this flawed design, though, because they had spent too much time developing this model and were locked into traditional schedule constraints. Additionally, the squads faced pressure from other sub-teams of MRDT that were anxious to begin testing their own systems. In the end, auxiliary systems were left unfinished until late April, resulting in rushed low-quality systems.

As a result, many Telemetry team members grew apprehensive of the traditional method and started advocating to implement the agile methodology. Thus, the team began to experiment with a blend of the two design philosophies. Since the implementation of a more agile project management style on the Telemetry sub-team, members are able to begin designing software and electrical hardware earlier in the year, which helps to ensure that their systems are ready for initial testing by the time mechanical systems are fabricated. While each philosophy works well within the confines of individual sub-teams, conflicts often arise as the Mechanical and Telemetry squads interact throughout the rover's development.

Strict, traditional schedules of Mechanical components clash with the loose, occasionally non-existent deadlines of agile projects, and team budgets are left incomplete without details about potential Telemetry expenses. MRDT faces these challenges as an outcome of its design process. With purely traditional development, designs are completed early in the project life cycle, whereas in pure agile development, designs are flexible enough that they can change throughout the project life cycle. For URC, the team begins with a basic design, clearly defined end goals, and a rough plan of how to complete all tasks. When only working towards a certain functionality, the final cost and image of the rover are uncertain at best. These numbers are not necessarily made clearer as the project progresses, either. When the number of revisions a project will need in order to achieve the desired end result is unknown, it is easy to over- or under-allocate funds to different sub-teams. For instance, three revisions of a printed circuit board versus four revisions can have a difference of several hundred dollars, and with a project cap of 15,000 USD, the team cannot afford to drastically over-allocate funds to certain systems.

This uncertainty presents more than just budgeting and scheduling challenges, however; it makes it incredibly difficult to allocate personnel. If the team does not know the full scope of one of the rover's systems, it is hard to determine the amount of personnel and the skill levels required to complete it. These types of instances can cause an uneven distribution of team members and experience levels across projects, which can lead to some systems falling behind others. This is exacerbated by MRDT being a volunteer organization, as team management cannot force a team member to be interested in specific tasks. These challenges in personnel allocation can prevent systems from being integrated into the rest of the rover, resulting in missed deadlines and rising tensions among team members who already clash over differing ideologies. Constant tension between members can create an uncomfortable work environment for both new and old personnel, and the intense support for one design philosophy or the other can give the inaccurate impression that the team does not welcome new ideas. If left unresolved, this image can drive away potential members and other opportunities for the team.

MITIGATION

The conflicts faced by implementing two separate design philosophies in tandem are unavoidable. However, these problems can be mitigated primarily through cultivating the interests and passion of team members, as harnessing this passion helps to promote compromise when it comes to defining the team's expectations, schedules, and budget. Providing an individualized approach to allocating responsibilities helps to foster excitement for and commitment to the team and its success. Potential team members are encouraged to move between squads until they are content, and specialized technical trainings are open to all team members. Additionally, the team attracts students across disciplines with shared interests, allowing members to network and form friendships with students of other ages and backgrounds. These aspects of MRDT create an inviting atmosphere conducive to accelerating the rover's design and development.

Maintaining this atmosphere and having the capability to make these compromises requires the team to develop a strong foundation of well defined goals, requirements, expectations, and responsibilities each year. Without such, it would be nearly impossible for team members to communicate effectively and work within the boundaries of two distinct design philosophies. To build this foundation, the team begins each rover's project life cycle with an intensive research and design phase. In the weeks following URC, a competition debrief is hosted so that the team can analyze its recent performance as well as its operation as a whole over the past year. Through this analysis, the team derives goals for the upcoming year and discusses how to adapt its organizational structure so that the team's endeavors are supported by its management practices. Choosing how to adapt, however, is often a cause of friction between team members, as the proponents of each design philosophy debate over which would be a better management choice for the upcoming year. To settle these discussions, the members of MRDT shift their focus back to the team's commonalities: its goals for competition and its overarching vision.

In order to achieve these goals, the team further develops its foundation by establishing clear expectations and delegating responsibilities before the design phase. This process begins by

explicitly defining requirements for the rover, which allows for more effective communication and execution of designs throughout the year. It is crucial to the team's success that each of these pieces are discussed and understood amongst proponents of both philosophies. After a thorough examination of current URC rules and regulations, the team begins to brainstorm different systems and methods for achieving success in each competition task. During this time, the team also defines all necessary systems, allocates responsibilities across each sub-team, and breaks interdisciplinary projects into manageable pieces. One year, for example, team members almost forgot to fabricate the rover's drill - a component essential to the Sample Return Task of URC - due to a miscommunication between sub-team leads. While the drill officially falls under the Science sub-team's responsibilities, its members lacked the mechanical experience required for the drill's design and fabrication. It became a situation in which the Mechanical sub-team believed that the Science sub-team was handling its fabrication, while the Science sub-team thought it was the Mechanical sub-team's responsibility. This mindset continued for a couple months before it was realized that no one was working on the designs. Additionally, no one had supplied the Telemetry and Controls sub-team with the information they needed to begin software and PCB design to incorporate the drill with the rest of the rover's systems. After a discussion amongst team leads, it was ultimately decided that the Mechanical sub-team would take control of the project while being advised by Science team members. If this error in task allocation went undiscovered, the entire system could have slipped through the cracks and missed completion. The team, therefore, has come to give this aspect of project management a higher priority than other mitigation techniques.

For reasons similar to those described above, the schedule of both philosophies must be determined early in the year. In the experience of MRDT, though, scheduling is one of the most difficult facets of blending the two philosophies, as it requires ideological compromises from proponents on both sides. To help the rigid traditional schedule adapt to the uncertain, flexible schedule that accompanies agile development, MRDT's Technical branch implements several basic milestones for all sub-teams throughout the year. These include the following: Critical Design Review, Drive Date, Auxiliary Operation Date, Rover Reveal, and Competition. Each milestone takes into account the school schedule, the nature of the projects undertaken, the available personnel, and the operation of the team's other three branches. This allows each sub-team to have an idea of when systems need to be operational while still being able to implement its preferred project management approach. The individual traditional and agile schedules come together through system dependencies. If, for example, a system requires software before it can be integrated into the rover, the software responsible for the core functionality has a deadline that is set slightly after the hardware's deadline, so that testing and debugging can occur before the entire system is integrated into the rover. Thus, parts of software can be written in parallel with the mechanical structure and electrical hardware design, rather than waiting to begin until after the fabrication of physical rover components. This practice allows architecture design flaws to be discovered and alleviated earlier in the project life cycle. Additionally, work on the rover's auxiliary systems begins earlier in the year, and should on-rover software require further revisions, new features can be added without delaying other systems or removing the rover's necessary functionality for testing. Each system is analyzed in this fashion to create the overarching schedule. Yet, the team's schedule is never completely finalized; it is instead evaluated and adapted on a weekly basis, providing the team opportunities

to account for unknowns rather than forcing members to cut corners in order to meet project deadlines.

To balance funds between sub-teams relying on different schedules and ideologies, MRDT first creates an overarching budget for the rover at the beginning of the year, based off of numbers from previous years and funds currently available to the team. As the design process progresses, a more detailed sub-team level budget is generated. The team waits to generate this sub-team level budget so as to allow the development of each system to drive the budget's creation, rather than solely using the budget to determine the rover's systems. To balance traditional and agile budget demands within the Technical branch, three revision opportunities are allocated for each Telemetry and Controls system. This practice helps available funds to flow easily between sub-teams as needed. If, for example, one system only requires two revisions, the remaining funds for the third revision can be reallocated to other systems. Team leads can also transfer funds from their sub-team to another sub-team more in need. Compromises such as these require students to look beyond their individual system and to make decisions according to what is best for the team. Overall, these practices act as a safety net to help ensure that all systems receive the funds necessary for their project's completion and have a chance for revisions. The team carefully documents all income and purchases, and carefully records how money is allocated among systems on rover. By using these flexible boundaries, MRDT accounts for the uncertainty associated with agile development, and helps to blend it with the thorough, steady documentation of traditional methodology. Adaptability is at the core of every mitigation technique described here, as the techniques must change as the team does to enable sustainability as well as advancement.

CONCLUSION

While there are projects better suited to the use of one design philosophy or the other, it is clear that blending traditional and agile project development can be a powerful management technique. Every member of MRDT at the Missouri University of Science and Technology has worked tirelessly to overcome the conflicts that accompany the implementation of these two philosophies. Regardless of which mitigation techniques the team employs to accomplish this, vision is the ultimate catalyst. MRDT operates under the vision "Today. Tomorrow. Forever." "Today" is the hard work that goes into building each rover. "Tomorrow" is the team's investment in its members. As students participate in the team, they gain valuable skills and experiences that will allow them to one day be the best possible engineers, scientists, managers, and leaders they can be. "Forever" is the team's commitment to giving back and furthering space technology. Most individuals involved in STEM have been inspired to pursue this field by someone or something. MRDT strives to be that someone for younger generations. This vision provides the energy needed to design, build, and test a rover and then compete with it every year. The team's continuing vision and yearly goals provide a common ground for the proponents of each philosophy, allowing team members to set aside their differences, communicate effectively, and develop a competition-worthy rover to the best of their abilities.

REFERENCES

[1] Mars Society, "University Rover Challenge", Lakewood, CO, http://urc.marssociety.org/, Accessed 2016, June 1.

[2] D. Hughey, "Comparing Traditional Systems Analysis and Design with Agile Methodologies: Waterfall Method", University of Missouri – St. Louis, 2009, http://www.umsl.edu/~hugheyd/is6840/waterfall.html, Accessed 2016, June 1.

[3] W. W. Royce, "Managing the Development of Large Software Systems", Proceedings IEEE Wescon – 1970, pages 1-9.

[4] T. E. Bell and T. A. Thayer, "Software Requirements: Are They Really a Problem?", TRW Defense and Space Systems Group – Redondo Beach, California, 1976.

[5] Agile Alliance, "Agile 101: What is Agile?", 352 Inc, Atlanta, GA, 2016, https://www.agilealliance.org/agile101/what-is-agile/, Accessed 2016, June 1.

[6] Agile Alliance, "Agile Glossary: Team", 352 Inc, Atlanta, GA, 2016, https://www.agilealliance.org/glossary/team/, Accessed 2016, June 1.

[7] C. Neagu, "Traditional and Agile Project Management", Stand by Soft, Craiova, Dolj, Romania, http://www.rationalplan.com/projectmanagementblog/traditional-and-agile-project-management-in-a-nutshell, Accessed 2016, June 1.

[8] M. Lotz, "Waterfall vs. Agile: Which is the Right Development Methodology for Your Project?", Seuge Technologies, Washington, D.C., 2013, http://www.seguetech.com/blog/2013/07/05/waterfall-vs-agile-right-development-methodology, Accessed 2016, June 1.