18 "Not-So-Successful" Case Studies

YOU CAN LEARN JUST AS much or more from companies that have attempted to implement an agile approach and failed or where it has not been completely successful. *Failure* in agile is seen only as a learning experience and is encouraged—without failure, there probably would not be much learning. Here are some quotes on the subject of failure:

Winston Churchill:

- "Success is the ability to go from one failure to another with no loss of enthusiasm.[1]

Thomas Edison:

- "I have not failed 10,000 times. I have not failed once. I have succeeded in proving that those 10,000 ways will not work. When I have eliminated the ways that will not work, I will find the way that will work."[2]
- "Show me a thoroughly satisfied man, and I will show you a failure."[3]

Joan Collins:

- "Show me a person who has never made a mistake, and I'll show you someone who has never achieved much."[4]

Well-known agile mantra:

- "Fail early, Fail often."

[1] "Miscellaneous Wit & Wisdom," National Churchill Museum, http://www.nationalchurchillmuseum.org/wit-wisdom-quotes.html

[2] Nathan Furr, "How Failure Taught Edison to Repeatedly Innovate," *Forbes* (June 9, 2011), http://www.forbes.com/sites/nathanfurr/2011/06/09/how-failure-taught-edison-to-repeatedly-innovate/.

[3] Laurence J. Peter, "Failure," *Peter's Quotations: Ideas for Our Time* (New York: Bantam Books, 1977), p. 177.

[4] Francesca Rice, "19 Joan Collins Quotes We Wish We'd Said Ourselves," *Marie Claire* (February 4, 2014), http://www.marieclaire.co.uk/blogs/545502/the-joan-collins-quotes-we-wish-we-d-said-ourselves.html. Several unverified sources claim that Albert Einstein said, "Anyone who has never made a mistake has never tried anything new."

The examples given in this chapter are companies that have had problems implementing an agile approach or the implementation has been incomplete. These are real companies; however, naturally, the companies are anonymous.

Agile can be a very difficult thing to do if a significant amount of cultural change is required. It is also a very empirical process, which means sometimes you have to try things to see what works and then make adjustments and corrections (a key idea behind agile is "fail early, fail often"). For that reason, these case studies should be regarded as learning opportunities and not failures. There's another well-known saying that is appropriate here: "If you have never failed at something, you're not trying hard enough." These companies should be applauded for trying to implement agile and, in many cases, they were ultimately successful after an initial false start.

COMPANY A

Background

Company A has a mid-sized IT organization. The company embarked on an agile implementation and trained most of the IT application development staff of about 80 to 100 people in agile practices. An agile coach was brought in for over a year to provide coaching to the teams, and the company made some progress on implementing an agile process at the development team level; however, due to cost-cutting pressure, the agile coach was let go, and there was little or no support at the executive level to take the agile process to the next level.

The approach

About 80 to 100 people in the organization were trained in agile practices, and, at least at a mechanical level, a number of agile practices such as daily standups were being implemented. However, the scope of the effort was very limited to the development organization; the approach was fairly mechanical without an understanding of the principles behind it and without much of an attempt to fit the approach into the company's business environment.

What went wrong

Table 18.1 shows what went wrong.

Overall conclusions

This is a great example of an agile implementation where the company was looking for a "quick hit" and didn't follow through enough to fully develop their agile approach. Agile is a journey and can require some significant organizational change to fully implement it. It is not an all-or-nothing

TABLE 18.1 Company A: What Went Wrong

Problem	Solution
Agile Is Not Just a Development Process	
The company's senior executives saw the agile process as having significant benefits to make IT development go faster; however, they saw it as an IT development process only and didn't see the benefits of investing beyond that level.	Many times companies see agile as a "silver bullet" that is going to make development go faster; they see it as an opportunity for a quick and easy "win" and not as more than a development process. This often results in a partial implementation of agile that doesn't take full advantage of the benefits it can provide.
On the surface, the development effort did *appear* to go faster, but the truth is that people were overworked to make it go faster, and the quality of the products really suffered as a result.	It is often necessary to start out with a limited implementation of agile to get started with, but it's important to set everyone's expectations early on that it is only a start and much more follow-on effort will be required to fully realize the benefits.
Commit Resources to Teams	
The company implemented many of the "mechanical" aspects of an agile development process (e.g., daily stand-ups were held), but there was no real change in the way people were assigned to teams. ■ People were not dedicated to teams and might be assigned to as many as three to four different teams. ■ In some cases, the developers didn't participate directly in the teams and were represented on the team by their managers.	This is, unfortunately, a common practice. Companies implement a few agile practices and call it *agile,* but it really is a very limited implementation of agile. On the surface, it looks "agile" because some of the agile rituals like daily standups are being followed; but it may be only superficial and not really consistent with the real principles behind agile. It is very difficult, if not impossible, to make an agile development process work effectively if the majority of the people on the team are not dedicated to that team and don't even participate directly in it.
This was clearly not consistent with a true agile approach. Without dedicated people on teams, it was almost impossible to stabilize the velocity of the teams and accurately predict performance.	The problem in many cases is that the company sees agile as a development process that only impacts the IT organization; focuses on the mechanical implementation level without understanding the principles behind it; and never follows through with a more complete organizational transformation to make it really work.
Change Is Essential	
There was no fundamental change in the way the company handled requirements: ■ A separate product management group was responsible for producing a business requirements document (BRD) and handing it off to the development team.	The company operated in a highly regulated environment within the financial services industry, and the control being exercised by the product management group had been seen as essential to tightly control and manage the product development effort. However, control is not an "all or nothing" proposition. There are lots

(*continued*)

TABLE 18.1 (*Continued*)

Problem	Solution
■ The process was not very collaborative and it was cumbersome. The product management group insisted on control of the requirements and became a "middleman" between the development team and the business users for clarification of requirement details.	of ways to implement an effective level of control over an agile project without overcontrolling it. The Managed Agile Development framework described in this book is an example.

proposition and sometimes requires a hybrid approach that is designed to fit the business environment. In this case, an agile coach was brought in and was successful in developing a foundation of basic agile practices, but it just didn't go far enough.

COMPANY B

Background

Company B is a mid-sized IT services company. The company has been rapidly developing a software application development business for their clients and the business has experienced significant growth. The company has used a waterfall process to manage its software development projects and bid fixed-price software development projects. The typical implementation of the process consisted of two steps:

1. There is typically a fixed-price effort to do the planning and design phase to define detailed requirements and design specifications for the project.
2. Following the fixed-price planning and design phase, another fixed-price project is proposed for completing the development and testing phase of the software solution.

Recently, Company B experienced significant problems with one of its largest customers. A critical project with this customer was completely stalled. Company B had accepted a fixed-price contract for delivery of the software based on some incomplete requirements and it wasn't making any significant progress, because Company B's development team was largely idle waiting for requirements to be further defined by the customer.

The customer had made a commitment to complete an initial installation of the software in several months and recognized that it needed to take charge of the situation to get the project moving. The customer replaced their project staff that was directing the project and brought in a new project manager and a number of business analysts to accelerate the requirements definition process.

The approach

After the customer replaced their project staff, the customer gave an ultimatum to Company B that the initial installation deadline still had to be met in spite of the earlier delays in completing the requirements. The customer then laid out a plan consisting of a number of sprints that it wanted Company B to meet in order to hit the initial installation date. The following are some of the most significant characteristics of this effort:

- Both the customer and Company B recognized that a more agile approach was needed to make progress; however, neither Company B nor the customer had any significant experience with implementing an agile software development approach.
- The methodology that was ultimately implemented was not really agile or waterfall—it was really just a brute force effort to get the work done in order to hit the deadline. It was similar to breaking up the overall project into a series of *mini-waterfalls* each being about two weeks in length that were called *sprints*.
- The customer created a development schedule for completing the project based on what it thought was needed to meet the installation schedule and broke up the functionality into sprints based on their estimates of the level of effort. Company B's development team was not directly involved in those estimates.
- Company B was pressured into making a fixed-price commitment for completing the project by the scheduled delivery date with performance penalties for missing the deadline based only on a very high-level understanding of the requirements.
- The business analysts that were brought into the project by the customer to accelerate the requirements definition effort worked with the customer's business stakeholders to create use cases and user stories to document the requirements and there was a very limited amount of direct communication with Company B's development team during that process.
- Prior to the beginning of each sprint, the business analysts who represented the customer turned over approved requirements documents in the form of use cases and user stories to Company B to be implemented in the next sprint.
- A limited amount of integration testing and user acceptance testing was included in each sprint due to the time pressures to get the work completed on time, and some time was reserved at the end of the project for doing final integration and testing prior to release.

What went wrong

In situations like this when there is a project failure, there is a tendency to take a brute-force approach to just put pressure on the situation to make the project work, rather than getting down to the root cause of some of the problems and taking a more systemic approach to address the core issues that

tend to make any project successful. In this case, there were four systemic issues that needed to be addressed, which are summarized in Table 18.2 in the following four general areas:

- Project Governance
- Process
- People
- Tools

In the end, the project turned into a "Death March" project to meet a firm delivery date that had been committed to with very incomplete requirements and, that in itself, was very problematic.

Overall conclusions

This is a great example of a project that was failing, and because of the time urgency of meeting a deadline, a brute-force effort was initiated to get the project moving without taking the time to take a more systemic approach to address the core issues that were causing the project to fail.

This is also a perfect illustration of the need for an agile project management approach. In this particular situation, a pure agile approach would not have provided much confidence of meeting the dates the project had to meet and a hybrid approach was needed that provided some level of predictability and control over the costs and schedule of the effort blended with a more agile approach for further elaborating the detailed requirements as the project progressed.

It also indicates the importance of a collaborative spirit of trust and partnership between the customer and the service provider to break down barriers to allow the project to work much more efficiently based on direct communications rather than an arm's-length contractual relationship.

TABLE 18.2 Company B: What Went Wrong

1. Project Governance	
Problem	**Solution**
From a business management perspective, Company B had not adequately recognized the risks and uncertainties associated with software application development projects and had not developed an effective business management approach for managing those risks and uncertainties. The company wanted the business with this customer very badly and very aggressively over committed to meet a fixed date with incomplete requirements.	Company B needed to redefine their management approach for managing software application development projects; however, given the time required by the customer to complete this particular project, it was not possible to do a reset and have a significant impact on the approach for this particular project. As a result, Company B had to take a brute force approach to get it done without a well-defined methodology. A better solution would have been to take an incremental approach to improving the management process as the project proceeded. That approach would have consisted of using an agile approach to identify and prioritize potential areas for improvement and implementing the most critical actions that could be done without significantly disrupting the project as it was in progress.

TABLE 18.2 (*Continued*)

1. Project Governance	
Problem	**Solution**
From a project management perspective, Company B had a PMO that was used for managing other types of projects but the PMO was not heavily involved in software application development projects and had no project managers who were trained and experienced in managing software development projects. A senior-level solution architect and a technical director were used to manage the effort for Company B. A project manager was assigned to play a supporting role to handle some project administration and reporting.	This is a great example of how the need for effective project management in a software development effort is often overlooked. The software development effort was seen primarily as an effort that needed to be led by senior-level developers—project managers were seen primarily as administrators who were more heavily associated with plan-driven, waterfall-style projects. Company B did not fully understand the concept of agile project management and how to better define the project roles to support an agile project management approach. In order to develop an effective approach, the roles in providing overall project management needed to be much more clearly defined. In an agile project some of the project management functions are distributed among the members of the team; however, in this particular situation, the team was not at that level of maturity; and, even if it was, that wouldn't necessarily eliminate the need for a defined project management role in managing projects of this nature.
2. Process	
Problem	**Solution**
Because there was no alternative to Company B's waterfall-style process for software application development, in any situation where that process doesn't work or isn't acceptable to the customer, the fallback was to use no process at all or be at the mercy of a customer-defined process from customers who are not sufficiently experienced to provide that kind of direction.	This is a classic case of where it is perceived that it is an "all or nothing" choice between a totally planned and controlled Waterfall approach and a totally unplanned and uncontrolled approach. A pure agile process would not have worked in Company B's environment as it would not provide a way of setting and managing customer expectations for the cost and schedule for completing projects. A hybrid agile approach was needed that provided a way of managing customer expectations combined with a sufficient level of flexibility and adaptivity to define the details of requirements as the project was in progress.

(*continued*)

TABLE 18.2 (*Continued*)

2. Process	
Problem	**Solution**
Somewhat of an adversarial relationship had developed between Company B and the customer because each side blamed the other for the earlier project failure. That made it difficult for Company B and the customer to develop a joint approach that was optimized to make the project successful.	In order to make this kind of approach work, it is essential to develop a collaborative spirit of trust and partnership between Company B and the customer to jointly manage expectations about the project.
3. People	
Problem	**Solution**
A more agile software development process was needed but that is heavily dependent on having highly skilled and well-trained people—it also can require a considerable shift in thinking and sometimes there is resistance to that kind of change.	In this situation, training of Company B's people was badly needed. However, a standard agile training course would have had limited effectiveness. An ideal solution would be to first better define how a hybrid agile approach would work and then provide training in the context of that approach; however, it wasn't practical in this particular situation to take that approach.
Attempting to do a project like this without people who are well-trained in implementing the project methodology is not a reliable, repeatable, and scalable approach.	A more pragmatic approach in this situation would be to incrementally implement process improvements and training as the project was in progress.
4. Tools	
Problem	**Solution**
In a fast-moving effort like this, tools can be essential for coordinating the efforts of the project team as well as tracking and reporting progress.	The may not be the most critical aspect of a solution for this situation; however, implementation of tools is one thing that can be done fairly easily and phased in without significantly disrupting the progress of the project and it can have a big impact by providing improved communication and visibility into project progress.
Because of the joint nature of the effort between Company B and the customer, the tool needs to be capable of sharing information openly and easily. However, that requires a spirit of transparency and openness for Company B and the customer to freely share information about the project.	Before a tool can be effectively used to share information freely and openly, there needs to be a collaborative spirit of trust and partnership between the customer and the service provider.

COMPANY C

Background

Company C is a company with a small IT organization that has been in the primary mode of supporting existing legacy applications. The company has not had to develop a major new application for a number of years. The existing legacy applications evolved gradually and were developed incrementally over a long period of time. The company initiated an effort to replace and redesign a large, existing legacy application and decided to use an agile approach for the development effort. The company had no previous experience with agile, and agile coaches were not engaged to provide training and mentoring of the project team.

The approach

The company implemented an agile development process from the "bottom up" within the development organization. The process was limited to a development process only, the business participation in the process was limited primarily to JAD sessions to define the requirements, and the role of the product owner was not fully implemented.

What went wrong

Table 18.3 shows what went wrong.

Overall conclusions

It is very difficult to transform an entire company overnight from a traditional waterfall approach to an agile approach.

- Many times, it is appropriate to take a bottom-up approach and start with implementing an agile development process without attempting to transform the higher levels of management in the company; however, when that is done, you shouldn't ignore the higher levels and leave a void in those areas that *isn't filled at all*.
- A hybrid approach such as the Managed Agile Development process is many times a good way to integrate an agile development process with a more traditional higher-level management framework as a first step until those higher levels can be addressed and transformed to a more agile approach. It's like training wheels on a bike—learning to ride a bicycle for the first time as a young child can be a terrifying experience; children tend to fall over quite often and have many accidents. Many times, training wheels are needed until the child gains a sense of balance and the confidence to ride the bike without them.

TABLE 18.3 Company C: What Went Wrong

Problem	Solution
Product Owner Role	
The company didn't understand the role of the product owner in an agile Scrum project and used a traditional model for managing requirements. IT was held responsible for the overall success or failure of the project in meeting business objectives, and the business role was limited to providing input to requirements through JAD sessions. When IT takes primary responsibility for a project of this nature, there is a relatively weak focus on defining the business value that the project should produce without a product owner.	In companies that are in the primary business of developing software products, this role is obvious; however, in a company that uses internal IT applications to manage their business, the strategic importance of those applications may not be appreciated, and the role of the product owner (which is really equivalent to a product manager) might not be understood. A Product Owner needed to be appointed and trained to fill that role; however, given the scope and complexity of this development effort, a better alternative might have been to outsource the whole effort rather than doing it internally at all.
Project Governance	
The company did not have a clearly defined governance model of how the project would be governed. As a result, ■ The right people at the right levels were not engaged in making the right decisions about the project at the right times. ■ Direction from different people was sometimes in conflict. ■ Some people were making decisions that they should not have been making. For example, some of the executives in the company were making detailed decisions about such things as UI screen designs. ■ Some decisions, such as defining the high-level business objectives the project needed to fulfill, were not being clearly defined by anyone at all. ■ The project was way behind schedule with no end in sight and senior executives had lost confidence in the project being successful.	On large enterprise-level projects, there is a need for inputs and decision making from a number of people at different levels, and those inputs need to be organized. A good project governance model should engage the right people at the right levels to make the right decisions at the right times about the project. For example, ■ Senior executives should be defining measurable business objectives that the project should fulfill and delegating more detailed decisions about how the design of the system will fulfill those objectives to others. ■ The managers who are responsible for the business processes should be defining the business rules of how their processes should work. ■ The users and stakeholders who use the system from day to day should have a key role in defining such things as screen designs to ensure that the system is usable. Without clearly defining these roles and responsibilities, there may be conflicts among people attempting to give direction, and some direction might be left out. In this particular case, a clearly-defined project governance model needed to be defined and implemented.

TABLE 18.3 (*Continued*)

Problem	Solution
Development Process	
The company assumed that the development effort could be accelerated by simply breaking up the development process into sprints and using an agile approach for managing the sprints. However, because the agile development process was not fully implemented with the business, the project-level planning and release-level planning that should have taken place was neglected. The result was that the development team was off-and-running developing code, but there wasn't a clear plan for how that code would be released and what the minimum functional requirements for a production release would be.	For large, complex enterprise-level projects, an agile process cannot be implemented only at the development level; there has to be some higher level planning processes associated with it or it is not likely to be successful. ■ Project-level planning is necessary to define a roadmap at a high level for how the product will be rolled out in releases. ■ Release-level planning may also be needed for how the features will be allocated to releases. In many cases, it is also essential to document a high-level plan to define what assumptions have been made about how the project will be rolled out so that there is consensus and buy-in to that plan from all appropriate stakeholders.
Quality Assurance Testing	
There were no formal QA testing resources on the project, and whatever testing was done was done on an ad-hoc basis by developers and business analysts without any formal QA test training. This can be a problem for many agile projects. It was assumed that formal QA was no longer needed, and people on the project such as developers and business analysts who did not necessarily have any formal QA training would perform testing. The result was that testing was very ad hoc without a plan and without well-defined, repeatable test cases to ensure an adequate level of test coverage was provided.	Testing is a science, if it is done properly, and requires people who have some skill in developing well-designed test plans and test cases. An agile development process does not totally eliminate the need for that. A large portion of the testing effort can be done by developers and business analysts, but there is still a role for formal QA testing, especially on large, critical enterprise-level projects. Rather than having a separate QA department perform that function, testers can be integrated into the team, but whoever performs that role on the team should have the appropriate testing skills.
Architectural Planning	
In this particular project, there was a significant risk associated with the cutover of the existing legacy system to the new system that needed to be planned.	One of the agile principles is, "Best architectures and requirements emerge from self-organizing teams." This project illustrates how that principle needs to be reinterpreted at an enterprise level. In this particular case, the risks associated with

(*continued*)

TABLE 18.3 (*Continued*)

Problem	Solution
■ There were some significant architectural decisions associated with how the two systems would coexist with each other for some period of time for the transition to be successful. ■ The architectural planning associated with that transition was not given a sufficient level of focus, and a solution to this architectural problem was deferred until well into the development process.	these architectural decisions were so great that they needed to be addressed and a solution planned early in the project. In this situation, none of the developers on the team had the level of expertise required to do the level of architectural planning necessary. Expecting the development team to perform this function without a sufficient level of focus and expertise just isn't realistic. A separate work stream might need to be created that is staffed by people with the right level of experience and focus to do the architectural planning in parallel with the primary development effort in the team. In many large enterprise projects, particularly ones that require multiple teams, a separate team is responsible for architectural planning and direction.
Project Management	
This is an example of a project where a hybrid approach is needed to blend some amount of traditional project management with an agile approach. In this particular project, a project manager was assigned; however, the company tried to implement a pure agile approach at the development level without a higher level of planning to perform some traditional project management functions such as planning a roll-out strategy, developing milestones, and performing general risk management tasks.	In more mature agile teams, some of these project management functions might be performed within the team, but they become especially critical on large enterprise-level projects. This is a perfect example of the need to fit the methodology to the project rather than force-fitting the project to a pure agile approach. It is important to make an assessment of the scope and complexity of the project and develop an approach that is appropriate to the project. In this particular project, a hybrid approach such as the Managed Agile Development approach is probably needed to provide a blend of traditional project management at the macro-level with a more agile development process at the micro-layer.

TABLE 18.3 (*Continued*)

Problem	Solution
Company Culture	
The culture in this particular company was very sales-oriented and also very heavily focused on operational excellence: ■ The individual users had sales goals that they needed to meet, and there was a significant amount of pressure to meet those goals. ■ The management approach in the company had a strong command-and-control orientation. As a result, there was a lot of top-down direction to the project without a sufficient level of delegation of responsibility and empowerment of the team.	It's very difficult, if not impossible, to change a company culture like that to make it more compatible with an agile development approach, and any change in company culture can take a significant amount of time to implement. The best approach is probably to make the senior managers aware of the impact of these cultural differences and make a conscious decision of how to mitigate their impact. A hybrid approach such as the Managed Agile Development process can be a good way to layer an approach that adapts an agile development process to a culture like this that is not fully compatible with agile.
Tools	
An agile project management tool was used on the project, but no one on the project team was fully trained in its use. As a result, the tool was not well utilized, and it was very difficult to plan and organize the project.	Tools are essential in most cases to manage large enterprise-level projects, and people on the project team need to be trained in their use to know how to use them effectively.

This is also a good illustration of the need for agile coaching and training in implementing an agile transformation. Although the company had some people on the project who were trained in agile and understood the mechanics of how to apply an agile process at a development level, there was no one on the team with a sufficient level of training and expertise to implement an agile process on a large, complex enterprise-level development effort such as this.