
# Iterative Project Management to Achieve Results

Megan Torrance


# Iterative Project Management to Achieve Results

Megan Torrance

22 21 20 19 1 2 3 4 5



ATD Press is an internationally renowned source of insightful and practical information on talent development, training, and professional development.

**ATD Press**
1640 King Street
Alexandria, VA 22314 USA

**Ordering information:** Books published by ATD Press can be purchased by visiting ATD's website at www.td.org/books or by calling 800.628.2783 or 703.683.8100.

Library of Congress Control Number: 2019943505

ISBN-10: 1-949036-50-2
ISBN-13: 978-1-949036-50-3
e-ISBN: 978-1-94903-651-0

**ATD Press Editorial Staff**
Director: Sarah Halgas
Manager: Melissa Jones
Community of Practice Manager, Learning Development: Eliza Blanchard
Developmental Editor: Jack Harlow
Production Editor: Hannah Sternberg
Text and Cover Design: Shirley E.M. Raybuck
Cover Design: Darrin Raaum

Printed by Color House Graphics, Grand Rapids, MI

# Contents

# Introduction

The first time the term Agile was used to describe an iterative development process specific to software was with the Agile Manifesto written in February 2001. The Agile process aimed to make it easier for software engineers, their teams, and their business sponsors to work together and be adaptive, resulting in a better product for the end user.

But the concepts underlying Agile have much earlier roots. Some argue that Agile traces all the way back to the 1620s with the development of the scientific method by Francis Bacon. A more commonly thought of starting point is the Plan-Do-Study-Act (PDSA) cycle developed by Walter Shewhart in the 1930s. PDSA, like Agile, is an iterative and incremental development methodology that was adapted and used to train hundreds of managers at Toyota in the 1950s (Rigby, Sutherland, and Takeuchi 2016).

In the 1980s and 1990s, with the explosion of software development as an industry, leaders continued their search for better processes. Studies showed that teams who worked together and continued to refresh their design and development processes created more successful innovations much more quickly than their competitors. Two of the people at the forefront of this work were Jeff Sutherland and Ken Schwaber, who created the Scrum method, named after a rugby move in which players pack tightly together and move as one in an attempt to gain possession of the ball. Scrum had the same goals that Agile ultimately would: finishing projects on time, under budget, and with fewer bugs. Sutherland and Schwaber were then involved in the creation of the Manifesto for Agile Software Development (the Agile Manifesto) in 2001.

## The Agile Manifesto

The Agile Manifesto shapes the work of Agile project management teams. Unlike other manifestos, this one is quite short but no less powerful. A mere 68 words, the Agile Manifesto lays out these core values:

> 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.

The Agile Manifesto reflects collaborative, practical values and a desire to approach project management in a way that focuses on people—both the people who make up the project team and the end users of the product.

What it means for L&D in practice is:

- listening to team members and stakeholders and changing the product's look, feel, and features in response to feedback and changing needs; being willing to revisit and repeat phases, such as design and development, following iterative implementations and feedback
- prioritizing delivery of a responsive app that performs the tasks a learner needs over a complete, perfect, beautifully formatted project scope, technical manual, and set of interface specs—or a set of detailed wireframes or storyboards imagining the potential (but nonexistent) app

- revisiting lists of deliverables as the project evolves rather than holding to (and billing for) each item on the list whether it is ultimately needed or not
- adjusting the schedule if a member of the team is reassigned or unexpectedly absent

Thanks to these values, Agile has since become ubiquitous amid any team or organization developing software. Beyond the four core values, Agile teams follow a set of 12 principles, which turn a short and sweet statement of intention into actionable directions. These 12 Agile principles are:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. Face-to-face conversation is the most efficient and effective method of conveying information to and within a development team.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.

10. Simplicity—the art of maximizing the amount of work not done—is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

These 12 principles can easily apply to the L&D world too. In the appendix, I've detailed how I adapted each one specifically for developing learning projects; check it out now, or reference it as needed throughout this book.

## My History With Agile

My own career paralleled the emergence of Agile in the software industry. In the 1990s, as a project manager at Andersen Consulting (now Accenture) and Arthur Andersen, I followed their Method/1: Plan-Design-Develop-Implement, with evaluation left for the next plan phase (Rifkin 1992). Glorious in its detail and rigid in its implementation, Method/1 and I had a rather stressful love/hate relationship. But brute force and long hours could overcome any project management deficiency when you're in your 20s and don't know any better!

After leaving the firm and starting my own consultancy around LMS implementation and e-learning development in the early 2000s, I abandoned the rigid project planning ethos because my work in instructional design was "so much more creative." And so I spent just as much brute force and long hours, just without a solid project plan.

As my company, TorranceLearning, grew and our client projects got bigger, our loose approach to project management became unsustainable. Our clients were still happy with the results, but our work-life balance was out of control. Midway through a project, we had no idea if we would finish on time, if we would have to write off hours we couldn't possibly bill the

client, or if we'd be able to keep up with a constantly shifting set of needs and requirements along the way. We needed to do something better.

By happenstance, my social and business networking circle included a lot of software developers, and by this time Agile was becoming the norm in our local tech scene. I spent time with Dianne Marsh of SRT Solutions, Helene Gidley of HSG Consulting, Rich Sheridan of Menlo Innovations, Marisa Smith of the Whole Brain Group, and Rob Houck of LearnShare, soaking up what they were doing on their projects. This was 2008. Each of these small businesses had their own approach to Agile. Their similarities were helpful foundations, while their differences inspired us to make our own adaptations for the instructional design space.

In late 2011, we realized these adaptations were quite extensive. Our business model and way of engaging with our clients was fully wrapped around our project management approach. We wondered if the extent of our changes still qualified us as using Agile. We decided to call it the Lot Like Agile Management Approach and named it LLAMA®.

LLAMA works for us. It works for clients. And we felt like we had something to share with our peers. The TorranceLearning team and I have been sharing this approach with fellow L&D professionals since 2012. By now, thousands of people have learned about LLAMA and adopted it in whole or in part to their work.

In the middle of writing this book, Susan Lord, a courseware developer and project manager who attended a LLAMA workshop at a conference wrote this to me:

> Hi! I just wanted to tell you I did my first Agile chart with Post-its and tape on my wall. . . . It is enormous but I am no longer drowning. I got my team on board and they can visualize what is needed. Thank you, thank you!

> It outlines our process flows, what milestone we are at, and what needs to happen to complete this phase. And what is wonderful is there was no bossing anyone around. Which I love! Everyone was in it! Fantastic!
>
> I also found out my manager was in your class last month. So we are now speaking the same language.

This quick exchange over LinkedIn sums up many of the appealing aspects of Agile and LLAMA: the clarity of visible project management, team engagement, work-directed teams, and a shared vision with teams, leaders, and their business sponsors. These aspects are within your reach too.

## Who This Book Is For

I've written this book for all the instructional designers, course developers, learning experience designers, and other professionals leading projects in the learning and development or training space who are looking to find a better way to manage their projects and deliver better results, on time and in budget. Essentially, a better way to work. Our industry is not steeped in a project management culture, yet nearly all the work we do is done as a project, with a defined start and end date and a deliverable to be produced. The model we've followed for a half century or so—the ADDIE model—no longer serves us in a do-more-with-less world of constant change.

Whether you're creating instructor-led facilitated experiences, virtual classroom training, e-learning, performance support, mobile learning apps, or advanced digital learning experiences, your work is somewhat like the work of software developers. And the approach outlined in this book borrows heavily from the Agile approach used the software industry.

## What's in This Book

The book opens with chapter 1, which lays out the case for using Agile in an L&D context. It highlights where the traditional waterfall approach to project management (ADDIE) fails to respond to changing demands. It also presents my Lot Like Agile Management Approach, which adjusts Agile in ways to make it a better fit for instructional design.

Then, part 1 describes the project kickoff and setting a project up for success with Agile. Chapter 2 guides you through planning the project kickoff, including who needs to participate and what do you need to cover during it. Chapter 3 covers how you should define the project's goal, particularly whether it should be training or performance focused. Chapter 4 delves into how to craft personas from your learner base, then how to select the primary learner on which your training will focus. Chapter 5 borrows the concept of user stories from software development to help you define scope. Chapter 6 takes a different approach to scope definition, one more suitable to instructional projects, and offers the Action Mapping technique, which you can use to identify key behaviors related to the goal, then map activities and content to those behaviors.

Part 2 moves into the routine of actually managing the project. Here, you'll learn how to define tasks and deliver iterations of the product, as well as establish a sustainable working rhythm with your Agile team. Chapter 7 shows you how to plan for an iterative project, including lining up the high-level arc of the project with your daily workflow while anticipating the unexpected. Chapter 8 details the challenges in estimating tasks, and then presents four rules for dealing with said challenges. Chapter 9 gets into the core component of an Agile project, the iteration; it makes the case for why iterative design works and presents ways to get it right. Chapter 10 digs into the rhythms that govern Agile projects as well as how to work well with subject matter experts and Agile software

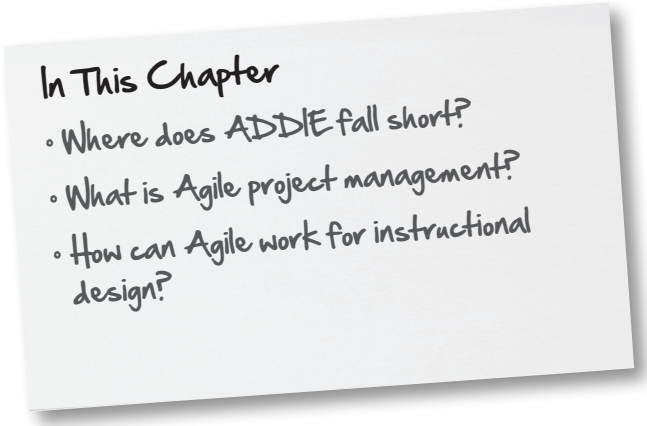
teams. Because open, regular communication is essential to Agile success, chapter 11 focuses on how you can ensure you're communicating in the right fashion with the right people. Chapter 12 examines the transformative power of the retrospective, both during iterations in the middle of the project and as debriefs once it's wrapped up.

Throughout the first two parts, the book discusses Agile as implemented on a single project. Finally, part 3 places Agile in a broader organizational context where multiple projects compete for attention. Chapter 13 shows you how to scale Agile beyond one project to manage and prioritize multiple Agile projects at once. Chapter 14 wraps up the book with a call to action for shifting the culture in your team, department, or organization to lay the groundwork for Agile. The appendixes contain ready-to-use job aids for applying the techniques in the book to your projects as well as a more detailed look at how each of the 12 principles of Agile can be applied to L&D. I recommend flipping back to it from time to time as you read and each principle comes into play.

Welcome to the world of Agile and LLAMA. I hope this book offers you the techniques and mindset for embracing a new way of working. Just as our projects are iterative and incremental when we use Agile, this method is as well. I welcome your engagement and feedback any time!

CHAPTER 1

# The Case for Agile




A woman approached the TorranceLearning booth at a conference several years ago.

She said, "Megan! I hear that you help people with their project management problems. I need your help."

I adjusted my cape, stood a little bit taller, and asked her about the problem.

She said, "You have to help me stop the 11th-hour changes!"

That made me pause a little bit. I wasn't sure how to respond.

She clearly didn't know that my whole project management "thing" was about accepting and expecting changes, even late in the project.

I asked her what she was making training about.

"Software."

I asked what kept changing.

"The software."


Was she really trying to stop the development of a product so that she could be on time and within budget with her part of the project? Even at the risk of delivering something that was wrong? Probably not. And yet the framing of her question—stopping change so she can finish her work—is probably familiar to many of us in L&D.

This anecdote illustrates the biggest problems with how instructional designers have managed projects for years. The focus has been on the wrong things: It's all about delivering something—anything—on schedule and within budget. Not that those are bad goals, but they leave a critical factor out of the equation: the learners. Your on-time, on-budget piece of training might not work. It might not do what the learners need. It might not meet the learning objectives.

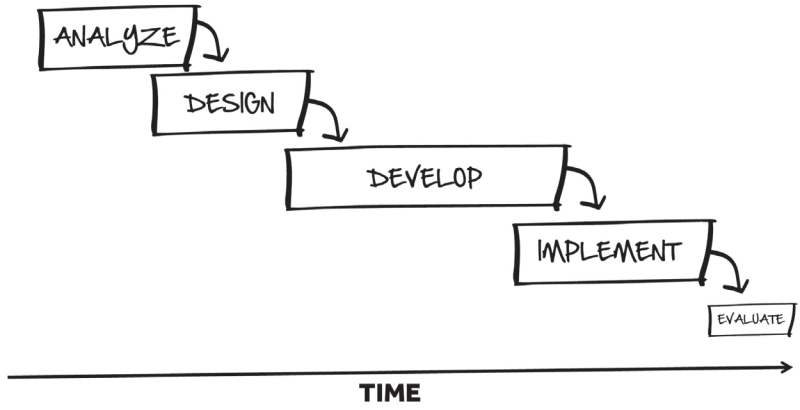
Let's put learners back in focus for our instructional design projects. But first, we need to clarify precisely why traditional project management methods are inadequate.

## What's Wrong With ADDIE?

The stalwart of learning and development project management is ADDIE, a decades-old linear or "waterfall" approach to planning and managing software and instructional design projects (Figure 1-1). ADDIE describes the five phases of project planning: analysis, design, development, implementation, and evaluation. While there's nothing inherently wrong with that formula, when applied literally ADDIE assumes a linear progression from one phase to the next. Once one phase is complete, the project team moves to the next phase. Generally,

there is no opportunity to revisit earlier phases; a developer can't climb back up the waterfall.

**Figure 1-1.** The ADDIE Workflow




In a model like this, if you go back to the drawing board it's because something went horribly wrong. Even more problematic, in a waterfall model, evaluation occurs only at the end. This leaves the end users' experience out of the design and development process. It also means that if problems or flaws in the design or implementation are found, they cannot easily be remedied. Problems wait to be fixed until the next product update, which could be months (or years) away, or they entail extremely costly changes, late delivery, and huge cost overruns.

One of my mentors used to say that the first day of a project is the worst day to plan what the end product will be, how much it will cost, and how long it will take to get there. ADDIE works fine . . . if nothing in your project changes from the day you draft the project plan to the day you deliver the training. But how often does that happen? Right: almost never.

For example, what if the technology changes during development? What if the target audience of trainees is assigned new or different goals,

and the learning objectives for the training change dramatically? As any instructional designer or developer knows, change is:

- scary
- frustrating
- inevitable
- happening faster than ever.

It's also:

- exciting
- an opportunity
- another word for improvement.

It's quite likely your project needs to change over time because the project requestor's or sponsor's needs evolve in tandem with the underlying business needs, as clients learn more about the learning experience or as ideas are tried and tested. To assume otherwise is to set yourself up for failure. It's folly to assume that the project sponsors know everything they want at the beginning of a project.

Rather than avoiding and fearing change or, like the desperate woman at the conference, attempting to prevent it, why not embrace change? Accept that it is inevitable, expect it in all your projects, and welcome changes as opportunities to make better products.

The truth of it is that everyone—the development team and the sponsors—is learning more about the project as it unfolds. And, quite frankly, it's often the instructional designers on the team who are coming up with new ideas as the project grows. (If you're not, you may not be fully engaged in the work you're doing!)

## ADDIE: A Relic of a Never-Existent Era

ADDIE hails from a bygone—and completely mythical—era when (if you just planned your project carefully and thoroughly enough) design, development, and implementation would progress smoothly, reaching

a scheduled, on-budget, happily-ever-after ending. Learners would get what they needed from the training, and the project team would cheerfully move on to the next neat, plannable project.

L&D professionals understand that to be just what it is: a fairy tale. While models like ADDIE can work in product manufacturing or construction, the linear waterfall model is inappropriate for product or learning development—or any innovative process. Processes with high variability simply cannot be pinned down in a plan written before design has begun.

Real-life project planning for training is a little bumpier than planning to manufacture countless identical products using a predictable process. Planning, designing, and developing L&D programs calls for the flexibility not only to adapt to change but to anticipate and welcome change, whether it's changing demands of project sponsors, changing preferences of learners, or changing business needs of organizations.

Only by testing incremental releases or partially complete products can you catch errors, clumsy features, and potentially disastrous problems early in the development cycle. By failing early, you can fix them relatively easily compared with the consequences of discovering a fatal flaw only when the final product is in the hands of hundreds or thousands of learners.

The solution? An iterative model like Agile project management.

## What Is Agile?

Agile is a team-based project management approach that emphasizes iteration and openness to change. An Agile team experiments and observes—and tests and gathers feedback on—a product as it is developed. Agile is ideal for projects where business needs might change, where specifications are not well defined at the outset, and where decisions are complex and require creativity. Agile builds in flexibility by:

- building deliverables in small increments
- releasing usable (testable) products multiple times during the development process
- applying feedback on the early releases to improve successive iterations.

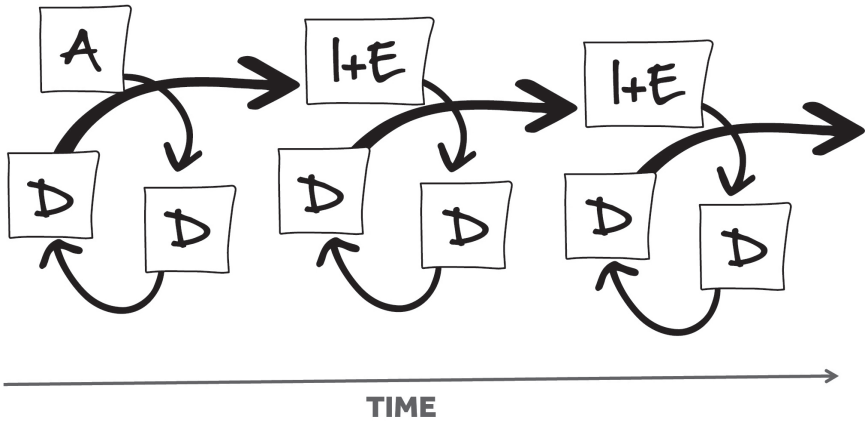
Before getting into how Agile translates to instructional design, let's start with my own concise definition: Agile is an iterative, incremental method of guiding design and building projects in a highly flexible and interactive manner, focusing on maximizing customer value and fostering high team engagement.

## The Lot Like Agile Management Approach

My definition of Agile fits in perfectly with my Agile approach adapted specifically for instructional design: LLAMA, or Lot Like Agile Management Approach. It is iterative and incremental (training to be tested, evaluated, and revised during design and development, rather than at the end). It guides design and build projects (remember Agile isn't an instructional design method itself and should not supplant best practices in that area). It is highly flexible (you need to be willing and able to respond to changes throughout the process). It is interactive (the team, the sponsor, and the subject matter experts work together). It maximizes customer value (your job is not to simply create training and move on to the next thing—you have to ensure your process delivers value to the customer, the end user). And it fosters high team engagement (whether you are a department of one or part of a multi-person function, you will need to engage a team of sponsors, stakeholders, subject matter experts, and learners to succeed).

This doesn't mean we need to throw away ADDIE entirely. The LLAMA approach includes the phases of ADDIE, with a twist (Figure 1-2).

**Figure 1-2.** ADDIE Adapted for Agile




Rather than assuming that the initial analysis covered everything and that no changes will be requested during design and development, Agile continuously returns to the design and development phases after successive evaluations. Rather than creating a single final product, Agile teams create multiple iterations. Projects are completed in small increments. In each phase, a product is created that stakeholders and learners can see, test, play with, and even break. This gives teams the chance to identify problems they hadn't anticipated or reevaluate features or functions that might not work in practice as they had envisioned.

It's also a way to accommodate changes that occur for reasons other than design errors. Maybe the end users' managers decided to buy tablets for all their sales personnel, and now the performance support tool has to work on mobile. Maybe a new product or a major upgrade demands additional training. Newly discovered information could render training methods or content obsolete.

The point is, the development team and the project management team cannot prevent change. And it's not possible to know everything about a

project in the initial design phase; nor is it reasonable to expect to anticipate all possible changes. What those teams can do is build in a way to respond to the inevitable changes.

Agile project management offers that flexibility and keeps the focus on the end user by emphasizing evaluation throughout. This phase, at the very end of ADDIE, often gets neglected. Here's how the evaluation discussion typically plays out, whether you're developing software or training: A project is done; it's been a long slog, but the team has delivered, finally! The last thing anyone wants to hear is what's wrong with the product. Any changes needed won't be implemented until the product is updated anyhow. So why spend time or money evaluating it?

With Agile for software development, evaluation is an essential part of each iteration. The feedback from the evaluation (the user testing or stakeholder review) is applied to the next round of design, development, and implementation. These cycles repeat until the product is done to everyone's satisfaction, until a hard deadline is reached, or until the budget is exhausted.

But, you might be thinking, project planning for an instructional design project is not exactly like project planning for a software product. That's true! It's important to highlight some of the differences:

First, planning an instructional design project requires a focus on learning objectives and desired performance outcomes, not just on software features and functions. An Agile team breaks up a development project into "user stories"—small, manageable units of work. L&D teams similarly break big projects into smaller units, but these are based on learning objectives. A unit of work might be a single learner activity or a content object. Each story or "task" card includes information on who is doing the task and how long it will take.

Second, unlike software development teams, which tend to be dedicated to a single project at a time, many L&D teams are working simultaneously on

multiple training and performance support products. This poses difficulties in planning that a dedicated software team is unlikely to encounter; however, the built-in flexibility of the Agile approach comes to the rescue here.

The Agile method involves breaking down large projects into very small pieces called stories. Team members estimate the amount of time each story will take. Thus a project schedule begins to emerge from these groups of stories.

LLAMA, like Agile, builds in tolerance for error. The time estimates are, after all, estimates. Schedules change; staff changes, gets sick, or goes on vacation. A task might take longer than expected. A needed content expert might be called away to deal with problems on another project. There's no way to plan for every contingency. But as long as clear, constant communication—an essential element of Agile and LLAMA—is the norm, this approach allows teams to get a realistic picture of who is needed when and for how long.

Third, L&D teams tend to consist of specialists; those who craft the learning objectives or provide the actual training content are often not developers. Even for digital learning projects, team members with software engineering and coding expertise might not have any instructional design knowledge. This specialization might lead to another key difference: The developers might have to wait for input from instructional designers or content experts at various stages in the development process.

That's where LLAMA comes in. Rather than trying to force an incompatible process to fit the Agile formula, LLAMA adjusts Agile in ways that make it a better fit for instructional design project management. In this book, I will extend the case I've just made for Agile for instructional designers by showing you how you can kick off the project the right way (part 1), then manage it through multiple iterations (part 2). By the end, I hope you will become your own advocate for why Agile makes sense in your organization!

## An Interview With Emily Ricco, Learning & Development Manager, HubSpot

**When and why did you decide to use Agile?**
I've been with HubSpot's Learning & Development team since 2014. When I recognized the opportunity for greater focus on learning design and content creation, I started an instructional design group. A few months after that, I adopted Agile because we had a long list of projects to tackle and very few people dedicated to tackling them. I wanted to empower this new team to take ownership of their work and how they got that work done. At the same time, I wanted to ensure we had strong communication and collaboration and could keep up with the pace of the business areas we supported. After doing some research, I discovered Scrum and related to the principles behind it: transparency, inspection, and adaptation. Two out of three of those principles are part of HubSpot's culture code, so it seemed a natural fit.

**What barriers did you have to overcome?**
I had to find the right version of Agile that would work for our team and our business. Everyone has their own opinions about Agile. Everything I read in articles and books and everyone I spoke with at other companies only increased the number of opinions to consider. Additionally, it was an entirely new way to work, and I had to strike the balance of avoiding micromanagement while ensuring consistent communication and structure.

**How does your organization's culture support Agile (or not)?**
Our organization's culture supports Agile because it is fast-paced and transparent. Agile allows us to set expectations and move quickly, which in turn allows us to be better partners. Transparency and adaptability are part of HubSpot's Culture Code, as well as part of the principles behind Scrum.

**Do you have a story to share about describing Agile to someone else?**
Our L&D team is mainly centralized but historically other teams have wanted to take charge of their own learning and development in order to move at a quicker pace. Last year, I used Agile as a way to build trust with one of those teams. I wanted to convince them that we were the right partners to work with on their ongoing training. When I described the opportunity they would have for providing feedback and the level of communication and transparency, the stakeholders were excited and confident in our ability to deliver. They even ended up providing headcount on our team from their budget so we could continue to partner with them and support them in a greater way.

## Key Takeaways

- A linear, waterfall-shaped approach to project management fails to incorporate the inevitable changes a project will face and sets up the project team for a struggle to manage those changes.
- An iterative, incremental approach like Agile accommodates change and offers a framework for meeting project needs while maintaining appropriate control of timeline and budget considerations.
- The Lot Like Agile Management Approach, or LLAMA, is an Agile approach adapted specifically for instructional design projects.

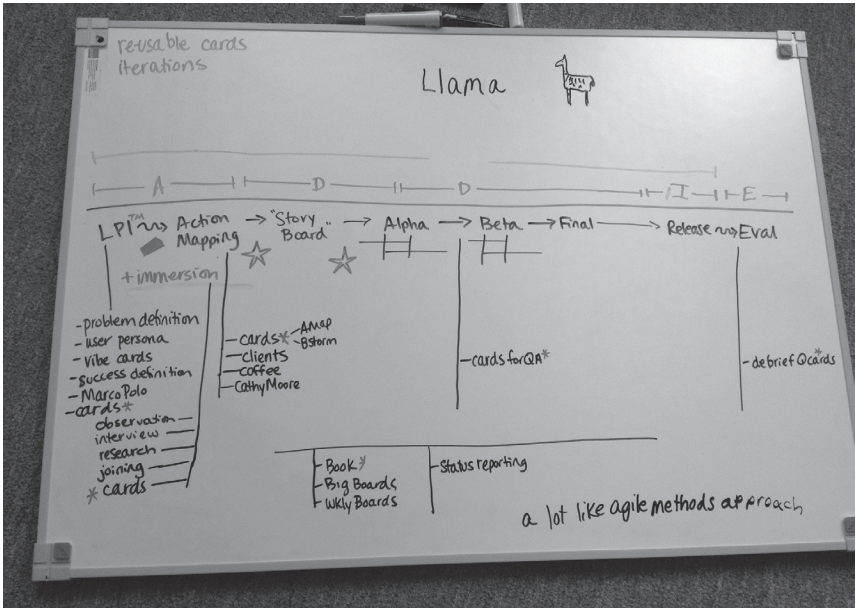
# Acknowledgments

This book is a culmination of more than a decade of practice, learning, and sharing.

The Agile Manifesto opens with the line, "We are uncovering better ways of developing software by doing it and helping others do it." This could not be truer of my journey with LLAMA, beginning by "uncovering" the secrets that my friends and colleagues in software development were using (thank you, Rob Houck, Dianne Marsh, Helene Gidley, and Rich Sheridan). It continued through years of "doing it" on projects with our clients who, in the early days, were patient with our growing pains (including Gary O'Neil, who at first declined to use our new "frou-frou" way of managing projects but ultimately became the supportive catalyst for so many of our ways of working), and who in recent years have seen Agile as an advantage. It's gratifying to now hear clients say, "We chose you because we know we're going to change our minds a lot and we knew that'd be OK with you."

The "helping others do it" is truly gratifying. A decade into our journey, thousands of people have attended a session, participated in a workshop, or read a book or article about our way of managing projects. (Had I known back then that this crazy idea of ours would take off, I would have started counting to have a more accurate number to capture the movement.) "LLAMAlumni" come from many countries on at least four continents, and I love getting photos and stories from people as they implement this way of working.

I could not ask for a better team of collaborators than the Torrance-Learning team, both current and past, who gamely give this a try, who wrestle with the implications of it, and who work with me to deepen the practice with their challenges and successes. Jen Vetter, Meg Fairchild, Alison Hass, Leanne Gee, Matt Kliewer, Dave Kerschbaum, Sue Kaba, Shannon Young, most recently Steve Wallag-Muno, and many others, have each had their hand in shaping LLAMA since this first rough whiteboard outline in the summer of 2012 when we gave it a name. We have built a business together based on Agile principles of flexibility, communication, sustainability, and transparency. Your spirit of generous collaboration (one of TorranceLearning's core values) is truly humbling and I appreciate your willingness to join me on this crazy ride.




A special thanks to Jack Harlow and Hannah Sternberg for your careful editing and patience with my Agile approach to writing a book, and to Shirley Raybuck for the interior design. And a shout out to all

the people who've encouraged me in the last few years to "write the damn book:" Marisa Smith, Catherine Juon, Cory Bouck, Michelle Massey Barnes, Kevin Daum, and Carla Torgerson . . . I finally sat down and did it!

Thank you to Zari Ozturk, Steve Wallag-Muno, and Matt Kliewer, whose graphical additions to the book have been essential in bringing the text to life.

Finally, I want to acknowledge the support from my family. Dad, your talent for writing about serious topics with a healthy dose of humanity and humor has been my inspiration throughout the writing of this book. Mom, thank you for your frequent trips to Michigan so I could be out on the road sharing LLAMA with others. And, Emily, yes, I'm doing the Skittles thing again. I love you.

# Appendixes

APPENDIX A

# The Agile Manifesto and 12 Principles for L&D Teams

## The Agile Manifesto

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.

## The 12 Principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Businesspeople and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. Face-to-face conversation is the most efficient and effective method of conveying information to and within a development team.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity—the art of maximizing the amount of work not done—is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

## The 12 Principles for Learning Development Teams

As we did for the values from the Agile Manifesto, it can help to ground the Agile principles in their application to instructional design. Here's how each one can apply specifically to L&D teams.

**Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.** This principle goes to the heart of iterative development. We don't take on a project, disappear into a black hole, and come back months later with a "final" product. We produce incremental "releases" of working products that the customer can try out. We're much more likely to get the end product right thanks to that input and feedback we've gotten throughout development. In the L&D context, we may release a large learning project in smaller phases rather than wait until an entire program is completed. Or, we may pilot a simple version of a learning experience in a fast-moving part of the organization to meet their needs quickly while learning from their results to apply to the rest of the organization.

**Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.** Software and L&D are both environments where changes should be assumed, expected, and accepted. By embracing this reality, Agile teams are able to meet changing needs, even if changes happen late in the development. The end result is therefore much more useful to the end users—the learners—and the organization.

**Deliver working [training] frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. Working [training] is the primary measure of progress.** These two principles go hand-in-hand and I often discuss them together. Delivering early, frequent "working drafts" of the final product gives you an opportunity to get feedback; it's also a hedge against shifting priorities and budget cuts. If you've delivered something, even if it doesn't have all the bells and whistles on it you'd like, you've delivered value even when you couldn't "finish" a project. For this reason, Agile L&D teams will tend to deliver work in modules, focusing on the highest priority ones or those most ready to move forward faster.

**Business people and developers must work together daily throughout the project.** We once had a client who had an incredibly aggressive timeline and almost no budget for a project we really wanted to do. The project was to develop concussion awareness training for Michigan's youth sports coaches. At the time, I was coaching my daughter's ice hockey team; we had a roller derby player as a course developer; and one of our instructional designers coached community flag football. I had had a concussion a year prior myself playing hockey.

*We all really wanted to do this project.*

It was going to be a challenge, though, because the subject matter expert was a physician with a busy schedule. So we moved the client's project manager into our office. We gave her a desk, the Wi-Fi password, and all the coffee she could drink.

While perhaps extreme, this arrangement meant that whenever we had a question, she was right there to answer it. It also meant that she was in tune with our workflows and our work processes, so, as things changed, we adjusted together.

Moving a member of the client's team into your office might not be workable (or ideal), but some framework for frequent communication and quick responses to questions is essential. The closer the communication and collaboration, the easier it will be for all involved—stakeholders and developers alike—to make needed changes.

**Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.** Many L&D professionals who attend my workshops find that this principle resonates. It seems as though they are not often given the opportunity to be creative in their work or to work in an environment where they are trusted to make decisions.

By working *with* the business rather than against the business, by accepting change instead of resisting or preventing change, and by

delivering frequently and delivering on time and in budget, Agile teams create an environment in which trust can be built.

**The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.** This one is a tough sell in today's distributed environment of global teams and home-office workers. While remote work is certainly possible and often efficient, face-to-face communication remains the richest communication channel we have. Where possible and where critical, it's often worth it to make the extra effort to be together as a team. We insist on kicking off our projects face-to-face whenever possible, and we have never once regretted it.

**Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.** No development process is perfectly smooth; Agile teams still have occasional late nights or long weekends. But planning by iteration makes it easier to manage the peaks and valleys of the work and maintain a sustainable pace.

**Continuous attention to technical excellence and good design enhances agility.** Agile project management supports, but does not inform, instructional design strategies. Solid instructional design and excellent project management are mutually reinforcing constructs; your projects need both. Remember, ADDIE is not an instructional design strategy either; it is a project management approach. Adopting an Agile approach does not infringe on instructional design strategies any more than using ADDIE does.

**Simplicity—the art of maximizing the amount of work not done—is essential.** With Agile, the focus is on meeting the needs of the customer. In L&D, that customer is the learner, the end user of the training. That focus is already a simplifying factor.

In the training development world, we tend to over-decorate and over-work some things. (You know you do it!) What's more, implementing

extras makes the product harder to change as underlying needs shift in the future. By over-decorating, we do more work now, and we create more work for ourselves in the future—perhaps with little to no benefit to the learners.

**The best architectures, requirements, and designs emerge from self-organizing teams.** Many years ago at a NATO e-learning summit, I met a retired German military officer who was doing some training consulting to NATO. He asked me if my team hated Agile as much as his team did. Hated Agile? We love it! Taken aback, I asked him to explain.

He described how he had created all of the tasks and required team members to put status stickers on each of them. He described setting all of the project's work estimates for the team, and he told me how he laid out all of their tasks for them each week. He then said that his team told him that he was micromanaging them.

That's not Agile project management.

The part that he missed in all of this is that, on an Agile team, members work together to manage the project. The person responsible for a task estimates the time needed, for example—not the boss. While there is often a project manager, the entire team is responsible for making sure that the work is planned.

**At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.** Agile teams call these discussions "retrospectives." In these conversations, the team examines not the product itself, but rather the process they used to get there.

## APPENDIX B

# Job Aids

## Project Kickoff Session Agenda

| Timing | Topic | Led By |
|---|---|---|
| 15 minutes | Introductions and Approach<br>• Introductions<br>• Agenda for the Day<br>• How We Use Agile | Project Manager |
| 1 hour | Define the Business Problem and Business Goal<br>• What is the business goal? How is it measured?<br>• What are the learners' goals? How are they measured? | Project Manager |
| 15 minutes | Break | |
| 1 hour | Define the Learner<br>• Create Learner Personas<br>• Select the Primary Learner Persona | Project Manager or Instructional Designer |
| 15 minutes | Break | |
| 1 hour | Define the Scope<br>• Action Mapping<br>• (Optional: User Stories) | Project Manager or Instructional Designer |
| 1 hour | Lunch | |
| 1 hour | Define Key Instructional Parameters | Instructional Designer |
| 1 hour | Define Key Project Parameters | Project Manager |
| 15 minutes | Break | |
| 30 minutes | Overall Project Budget and Timeline | Project Manager |
| 30 minutes | Iterations and Review Responsibilities | Project Manager or Instructional Designer |
| 30 minutes | Wrap Up and Next Steps | Project Manager |

## Learner Persona Questions List

Here are some sample questions to use as idea starters when creating a learner persona. Don't feel compelled to answer them all, as some will be more important to your project than others.

### Demographics

- What is his/her name?
- What is his/her age?
- What is his/her gender?
- What is his/her primary language?
- What is his/her educational background?
- What is his/her marital and family status?
- Where does he/she live?

### Professional Demographics

- What company does he/she work for?
- What is his/her job title?
- How long has he/she been with the organization?
- How long has he/she been in his/her current position?
- What is his/her income?
- How long has he/she been in the field of work?
- What job responsibilities does he/she have?
- Who does he/she interact with daily?
- How many people does he/she oversee?
- How much training and experience does he/she have with this topic?
- What end goals does he/she have related to the course?
- What experiences does he/she hope to have while taking the course?
- What career aspirations does he/she have?

- How does he/she feel about his/her current role with the organization?
- How often does he/she change jobs or employers?

Connection to the Learning and Learning Approaches

- What motivated he/she to take the course?
- How often will he/she apply what he/she learned in the course?
- Has he/she taken training courses before?
- What type of computer does he/she typically use?
- What is his/her comfort level with computers/technology?
- What is his/her comfort level with the topic of this course?
- What other skills and abilities does he/she have?
- Does he/she have any disabilities or health problems?
- How open to change is he/she?
- Where will he/she take the course?
- What other types of professional development does he/she do?

Away From Work

- How does he/she use social media?
- What does he/she like to do during spare time?
- What was the last book he/she read?
- What car does he/she drive?
- What is his/her favorite music?
- Where does he/she eat lunch?
- Is he/she more introverted or extroverted?
- How often does he/she exercise?

## Learner Persona Template

# LEARNER/USER PERSONA

Name

Image

Demographics

Technology skills & preferences

Job & roles

Favorite brands, trends, culture

Motivations & comfort zones

Experience with this concept

Motto

# Written Status Report Outline

**STATUS REPORT:** Project Name

**Overview**
(paragraph outlining the week's status in a nutshell)

**Activities We Completed Last Week**

- Task 1

**Activities Planned for This Week**

- Task 1

**Issues to Resolve/Open Questions**

- **Green:** does not hinder our progress in the coming week.
- **Orange:** could hinder our progress in the coming week.
- **Red:** we are on full stop and cannot progress without resolution of the issue.

Issue 1 Description

**Estimate-to-Actuals**
(graph and paragraph discussion)

# Capturing Retrospective Feedback

### Proud Moments

### Embarrassed Moments

### Budget/Timeline

### Start-Stop-Continue

| START | STOP | CONTINUE |
|---|---|---|
| | | |
| | | |
| | | |
| | | |

### To-Do Items

# References


Beck, K., M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas. 2001. "Manifesto for Agile Software Development." http://agilemanifesto.org.

Chapman, B. 2010. *How Long Does It Take to Create Learning?* Chapman Alliance LLC.

Cohn, M. (N.d.) "About Mountain Goat Software" www.mountaingoat-software.com.

Defelice, R., and K. Kapp. 2018. "How Long Does It Take to Develop One Hour of Training? Updated for 2017." TD.org, January 9. www.td.org/insights/how-long-does-it-take-to-develop-one-hour-of-training-updated-for-2017.

Moore, C. 2017. *Map It! The Hands-On Guide to Strategic Training Design.* Montesa Press.

Reinerstein, D., and S. Thomke. 2012. "Six Myths of Product Development." *Harvard Business Review,* May. https://hbr.org/2012/05/six-myths-of-product-development.

Rifkin, G. 1992. "Andersen Consulting's Culture of 'Clones.'" *New York Times,* September 6. www.nytimes.com/1992/09/06/business/andersen-consulting-s-culture-of-clones.html.

Rigby, D., J. Sutherland, and H. Takeuchi. 2016. "The Secret History of Agile Innovation." *Harvard Business Review,* April 20. https://hbr.org/2016/04/the-secret-history-of-agile-innovation.

Russell, L. 2016. *Project Management for Trainers,* 2nd Ed. Alexandria, VA: ATD Press.

Torrance, M. 2014. "Agile and LLAMA for ISD Project Management." TD at Work, November. www.td.org/td-at-work/agile-and-llama-for-isd-project-management.

# About the Author

Megan Torrance is CEO and founder of TorranceLearning, which helps organizations connect learning strategy to design, development, data, and ultimately performance. Megan has more than 25 years of experience in learning design, deployment, and consulting. Megan and the TorranceLearning team are passionate about sharing what works in learning, so they devote considerable time to teaching and sharing techniques for Agile project management for instructional design and the Experience API. TorranceLearning hosts the xAPI Learning Cohort, a free, virtual 12-week learning-by-doing opportunity where teams form on the fly and create proof-of-concept xAPI projects.

Megan is the author of *The Quick Guide to LLAMA* and two *TD at Work* publications: "Agile and LLAMA for ISD Project Management" and "Making Sense of xAPI." She is a frequent speaker at conferences nationwide. TorranceLearning projects have won several Brandon Hall Group awards, the 2014 xAPI Hyperdrive contest at DevLearn, and back-to-back eLearning Guild DemoFest Best-In-Show awards in 2016 and 2017 with xAPI projects. TorranceLearning is a 2018 Michigan 50 Companies to Watch.

A graduate of Cornell University with a degree in communication and an MBA, Megan lives and works near Ann Arbor, Michigan.

# Index

## M



## O

## P

## R



## S

## T



## U



## V



## W

BOOKS

# Agile for Instructional Designers

Iterative Project Management to Achieve Results

By Megan Torrance

## Pricing

| FORMAT | MEMBER | NON-MEMBER |
|---|---|---|
| Paperback | $24.99 | $29.99 |

**Please see the terms of sale for important information about the use of PDFs.**

1 QTY

Pre-Order

Save when you buy in bulk.

Available Tuesday, August 27, 2019

### Discover Agile for Better Instructional Design

To serve business needs amid greater volatility and uncertainty in the workplace, learning and

development professionals need project management methods that can keep up. Enter Agile.

Popular in the software development space as an approach to project management, when applied to instructional design, Agile provides a framework for adapting to change as it happens and for delivering the content most needed by learners. *Agile for Instructional Designers* proposes using Agile methodology to manage training projects and highlights where traditional linear processes have failed the business and the end users.

Recognizing that software development and instructional design have different needs and outcomes, author Megan Torrance developed the LLAMA methodology. Her approach adapts the common phases of ADDIE to incorporate the incremental, iterative nature of Agile projects. It allows learners to test and evaluate which features or design functions work before they're finalized. It also offers a way to accommodate inevitable mid-project modifications pushed by stakeholders, subject matter experts, or organizational leaders.

With templates for goal alignment, learner personas, scope definition, estimating, planning, and iterative development, *Agile for Instructional Designers* is the resource you need to embrace change in learning and development.

Read More

Book Details
ISBN: 9781949036503
Pages: 208
Publication Date: August 2019
Formats: Paperback
Product Code: 111910

## About the Author

Megan Torrance

Megan Torrance is the chief energy officer of TorranceLearning, an e-learning design and development firm outside of Ann Arbor, Michigan. She has spent over two decades knee-
Full Profile

See All

## Reviews

Karl M. Kapp, Ed.D.
Professor of Instructional Technology, Bloomsburg University

Megan Torrance has written an exciting, engaging, and immensely informative road map for applying an Agile mindset to the design of instruction. If you are developing any type of instructional project within your learning organization, this is the book that will enable you to finish your projects on time and within budget.

evin-Siegel.jpg

Kevin Siegel
Owner and Founder of IconLogic, Inc; CEO and Founder, International Council for Certified Online Training Professionals

Megan Torrance is a rock star in this field. She tells it like it is, and in terms all of us can understand. If you're looking for proven project management methods that help you keep your sanity, stay on time, and stay within budget, this book is a must-own!

## Be the first to comment

Sign In to Post a Comment

Sign In