

Extracted From:

Creating Your Dojo

Upskill Your Organization for Digital Evolution

Dion Stewart and Joel Tosi

Copyright © 2019 Dion Stewart and Joel Tosi

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

CREATING YOUR

DOJO

Upskill Your Organization
for Digital Evolution

DION STEWART & JOEL TOSI



LIONCREST
PUBLISHING

COPYRIGHT © 2019 DION STEWART AND JOEL TOSI

All rights reserved.

CREATING YOUR DOJO

Upskill Your Organization for Digital Evolution

ISBN 978-1-5445-0439-1 *Paperback*

978-1-5445-0440-7 *Ebook*

INTRODUCTION

Long-term commitment to new learning and new philosophy is required of any management that seeks transformation. The timid and the fainthearted, and people that expect quick results, are doomed to disappointment.

—W. EDWARDS DEMING, *OUT OF THE CRISIS*

“The coupons still aren’t working. The test cases are failing. Your code is wrong!”

“Your testing script is wrong. The code is written to spec. If you have a problem, go talk to the business!”

Joan, the payments team leader, rose from her desk and went to see what the commotion was about. One of her team’s developers was facing off with one of the test engineers in the hallway. The test engineer rolled his eyes and shook his head at the developer. When Joan asked them

what the problem was, the tester said the new coupons still weren't working correctly, but that the developer kept rejecting defects in the defect tracking system. They were at an impasse.

Joan was questioning her decision to accept the promotion to team leader she'd taken a couple months ago. She had been up late the previous night with her developers and people from operations trying to get their code deployed in the staging environment. Now it appeared the code was wrong and she might have another late night ahead of her doing another deployment. That assumed they could get one of their business partners to clarify how the coupons should work and get the code changes made today. As if that wasn't enough, she had a meeting in the afternoon with the stakeholders to explain why they were so far behind schedule. At a minimum, they were looking at another day of delay.

She was extremely frustrated. The codebase her team had inherited made it difficult to add new functionality with any kind of code quality. They rarely met delivery deadlines. Even though her company had sent hundreds of people through Agile training that advocated for cross-functional teams, her part of the organization was still siloed, with separate development teams and test teams. She couldn't put together a team with the skills she knew would address at least a few of their problems.

Even worse, new features weren't having the impact in the marketplace the business wanted. Her developers wanted time to clean up technical debt in the codebase, but the business just wanted more features faster, hoping they'd somehow achieve product/market fit. On top of all that, the organization mandated that all the services her team owned had to be moved to the cloud by the end of the year.

Something had to change.

Joan believed in her team's abilities, but she wasn't sure how they could make improvements with all the pressure they were under to constantly deliver. A few of her developers had attended a test-driven development course but that didn't seem to go anywhere because they'd had a hard time applying what they learned to their legacy codebase. There were several other two-day training courses her team members could attend, but she was skeptical there would be any real long-term impact.

She'd heard rumblings that her company had recently hired a new director of IT transformation and that he was creating something called a dojo to help teams get better at building digital products. She decided she'd look into it.

Does Joan's story sound familiar? Read on.

WHAT'S A DOJO, YOU ASK?

A dojo is an immersive learning environment within an organization where product teams learn new skills and new ways to solve problems. Teams apply those skills immediately to their work in the context of their real-world constraints and challenges. “Dojo” is a Japanese term for meditation halls and martial arts studios. It translates literally as “place of the way.” In the tech industry, dojos are pointing the way to better products and methods, and more effective problem-solving and learning within organizations. Dojos are leading organizational and cultural transformations.

Unlike traditional training where individuals or teams go to a conference or seminar, or external consultants come into the organization to conduct workshops, a dojo is a space unto itself within an organization, run by dedicated employees of the organization. It goes beyond the in-house training centers that teach one-off skills.

A dojo is an investment for the organization—in people, space, and time. For small organizations (with, say, less than four or five product teams)—a physical dojo space may be overkill, but the approach to learning presented in this book still applies.

For large organizations, an investment in a dojo starts with an honest conversation around whether a dojo

can meet the needs of the organization. This ultimately comes down to whether there is executive-level support for evolution through learning. If your leadership believes in investing in your people and giving them time to improve their skills, then you are ready to start looking into a dojo. On the other hand, if leadership believes in a guaranteed transformation through large consulting firms with fancy frameworks telling you what to do—well, history tells you how that story ends.

When an organization invests in a dojo, it becomes an attractor for new talent. Perceptions about the company shift within the development community. People recognize the company is doing something new and interesting. More importantly, it becomes clear that the company is investing in developing their people.

If you decide to add a dojo to your organization, you probably have people who can fill the various roles in your organization already. You might have to hire outside consultants in the beginning to get your dojo up and running. But let's not get ahead of ourselves—you'll learn the answers to all your questions in the upcoming chapters.

YOUR NEXT QUESTION IS, WHO ARE WE?

We've been working in the learning and digital product space for the past two decades as both developers and

coaches. We've been helping organizations set up and launch dojos since 2014. The examples you'll read about are culled from our experience, although we've changed details to protect the identity and privacy of the companies involved.

Our approach involves solving problems holistically and isn't limited to writing better code and using better technology. It includes focusing on creating better products and fully engaged teams, all while fostering deeper, stickier learning. We believe the challenges to creating great digital products go beyond technology. When we work with teams, we address practices across the entire product development value stream.

A HOLISTIC APPROACH

Teams are faced with a conundrum: they often want to reduce delivery times, improve technical quality, and create products with a better product/market fit, but they must slow down to learn the practices that will improve their product development capability.

The dojo takes a holistic approach, bringing together cross-functional teams and placing them in an immersive learning environment. Teams learn new practices and how these practices tie together. Unlike traditional training where practices are learned in sandbox environments

with canned examples, teams learn within the context of doing their real-world work, where constraints come into play. Learning becomes part of what teams do, as opposed to a special event.

THE SIX-WEEK DOJO EXPERIENCE

Joan sought out the director of IT transformation, who put her in touch with us. In our early conversations with Joan and her leadership, we backed her up in her desire for creating a cross-functional or full-stack team—at least for the time they'd be in the dojo. This new team would have all the necessary people to deliver a digital product. We guided Joan to assemble a team that comprised not only developers and testers, but people from the business side of the organization who understood the product being developed, designers, and people from operations.

Although the team members were all employed by the same organization, they had never actually worked together as a unified team. Coming into the dojo was going to be different—the team would work together all day, every day, for the next six weeks.

We began the first day by starting to work with product discovery practices. This surprised them. They had expected the dojo to be another training where they learned about technical practices or focused on process

minutiae. The team member from operations couldn't understand why he was being asked to participate in product discovery. The whole team was skeptical about our methods. We asked them to indulge us for a few hours, and if they didn't find the product discovery practices valuable, we'd jump into technical practices.

Within the first hour, the team started a discussion that clarified their understanding of the product needs. For perhaps the first time, the domain experts, engineers, and testers were having a direct conversation with each other. Even the operations person got in on the action. For some of the engineers, it was the first time someone asked them to think about the people using the product instead of coding requirements from a specification. Discrepancies surfaced about expectations for the product they would build over the next six weeks. The team quickly understood the importance of doing discovery and design—together—before jumping into delivery, and everyone could see the impact taking this first step would have on the quality of their product. Needless to say, they found value in what they were learning, so we spent a week on product discovery practices.

We built a product backlog—but in a way that was new for the team. Instead of getting overly focused on creating a list of things to build, we focused on the outcomes they wanted to achieve for their product and the learning

they wanted to achieve while they were in the dojo. The team learned how to use story mapping¹ to give additional dimensions to the information. This supported the team’s ability to make decisions about their next best investments in delivering value with their product and achieving their learning goals.

At the beginning of the second week, we started on product delivery. The developers learned how to code using test-driven development, the team began learning how to do automated builds and deployments, and the test engineers learned how to automate tests that had previously been exercised manually.

At one point, the developer who’d been on the receiving end of the tester’s wrath spun around in her chair to talk to the tester. She said, “Hey, I know we’re still working on getting your automated tests to run as part of the build, but that functionality that you’ve been writing tests for is now deployed in the test environment. You can go ahead and test it.”

The tester pulled up his automated test cases, clicked a button, and ran all the test cases. When he saw that one of them was failing, he spun his chair back to the developer and said, “Hey, there’s one test failing. I’m not getting the

¹ To learn more about story mapping, we suggest Jeff Patton, *User Story Mapping: Discover the Whole Story, Build the Right Product* (Sebastopol, CA: O’Reilly, 2014).

expected return values.” The developer looked at the test results and said, “I know what’s wrong. I can fix that right away.” The code was fresh in her mind because she’d just worked on it.

Communication happens in real time in a dojo, and product team members learn how to collaborate effectively with each other. In this situation, the developer went back to the code, fixed it, and fifteen minutes later there was a new build deployed in the test environment. The tester ran the tests, and this time all the tests passed.

Joan smiled as she watched this exchange. Two former adversaries were now on the same team.

During the team’s six-week dojo experience, both the leader and the team saw improvements in their delivery times and in the quality of their product. They were able to make these improvements through applying new skills they learned in the dojo in the context of building their own product. Their improvements had a positive impact on the team and on the organization as a whole.

They could finally get the outcomes they had failed to achieve with two-day courses and workshops. Joan was already thinking about bringing her team back into the dojo later in the year to learn how to migrate their services to the cloud.

Would you like to offer your teams a similar experience?
You can.

LEARN TO CREATE YOUR OWN DOJO

The world today is increasingly fast-paced and is even more so for anyone who works with digital products. Change is the only constant. Organizations can no longer afford to send their employees to two-day trainings for learning that doesn't stick. They need to become learning organizations, where learning while doing is the norm, not the exception. To that end, organizations are using dojos to offer immersive learning experiences in-house.

In the pages that follow, you'll see how immersive learning in the dojo setting is far more effective than traditional training and coaching. We'll take you through the steps of creating your own dojo and offer guidance on the following topics:

- Aligning learning outcomes with your organization's product strategy and technology direction
- Choosing the practices you will teach
- Setting up the physical space
- Staffing a dojo
- Running six-week dojo experiences and other dojo offerings

- Leveraging a dojo to make improvements across the entire organization

This book is not a typical boring training manual or a treatise on technical practices. It's about creating real-world learning experiences, illustrated with stories and actionable tasks.

The concept of the dojo as an immersive learning and practice space goes back thousands of years in the realms of Buddhism and martial arts. Likewise, immersive learning is not new to the technology space. In the early days of Agile, several organizations offered immersive learning experiences. More recently, the DevOps community has embraced creating learning organizations. In the tech world, the dojo can be the first step for an organization to create a culture of continuous learning and improvement.

The first dojo opened in one company in 2015. Over the course of five years, more than thirty organizations we know of have implemented dojos in one form or another. Ten of them rank in the top one hundred of the Fortune 500. Half of the organizations are listed in the Fortune 500 and Fortune Global 500. They represent a broad range of industries including automakers, airlines, financial services, insurance, healthcare, retail, food and beverage, farm machinery, manufacturing, and telecommunications.

In addition, an informal working group called the Dojo Consortium meets virtually every two weeks and has participation from more than twenty organizations. They share information with each other on topics ranging from getting stakeholder support for creating dojos and hiring coaches to measuring a dojo's impact. We organized the first Dojo Consortium conference in Minneapolis in the spring of 2019. The consortium members have voted to make that a yearly event.

Dojos are investments in your people, your culture, and your organization. Dojos take time, but the results are impactful—happier and more engaged employees, better products, less organizational friction, and more satisfied customers.

You can't just hire your way into these outcomes. You can't hire the best people and put them in a broken system. If you create a dojo, as teams go through the experience, the cracks in your organization will begin to show—and that's an opportunity for your organization to grow and improve. You need to start today to create an environment that helps your teams and your organization learn.

Are you ready to take the first step into the immersive learning experience of a dojo? If so, turn the page. You and your company will be glad you did.

Chapter One

HOW WE LEARN

A single intense, out-of-context classroom event can only get you started in the right direction, at best. You need continuing goals, you need to get feedback to understand your progress, and you need to approach the whole thing far more deliberately than a once-a-year course in a stuffy classroom.

—ANDY HUNT, *PRAGMATIC THINKING AND LEARNING: REFACTORIZING YOUR WETWARE*

Once upon a time, you learned a trade, maybe through an apprenticeship, and then performed that trade more or less in the same way throughout your professional lifetime. If you wanted to be a blacksmith, you worked alongside a seasoned blacksmith until you could forge iron with a certain mastery. Your blacksmith education ended there. Neither the tools nor the materials changed. You honed your blacksmithing skills by the simple nature of repeating the same movements, but you weren't required to learn new skills.

In most professions today, whether you're a developer writing code for an app, a radiation oncologist treating cancer patients, or a chef cooking dinner for heads of state, you're required to learn how to use new tools as they reach the market and add new skills as practices in your field change.

Even if you manage to keep up with all the changes in technology and the technical skills associated with them, building digital products requires ongoing contact with your customers to stay current with their needs. As it turns out, there are practices for doing that as well. Adopting and implementing these practices helps to ensure your product maintains product/market fit over time.

Learning new tools and skills, then, is a never-ending requirement of modern professions. We must become lifelong learners, and the organizations in which we work must become learning organizations that promote and support ongoing learning.

In its most essential form, the dojo creates an environment where teams become lifelong learners in a learning organization. To understand why this is a more effective way to learn, let's consider why traditional training methods are ineffective.

YOU TRAIN A DOG, YOU TEACH PEOPLE

When you take your dog to a training school, he learns to do things by command: sit, stay, roll over. Training programs for humans often follow the same rote repetition. Humans, however, learn through experience and reflection. Following instructions isn't learning per se, and successfully duplicating steps in some recipe-like format doesn't mean we can reproduce the results on our own. To learn to consider a problem and solve it, we need to understand what's happening when we do the tasks.

Short-term training has a host of problems, but perhaps the most significant is that the training environment is never the same as the real environment (much like dog training, where there are never pesky cats on the training field to distract your dog from a long stay command). In a training sandbox, you're given explicit instructions—do this, do that, do the next thing—and nothing goes wrong. As soon as you return to your desk, you're dealing with constraints in the form of existing legacy code, security issues limiting access to resources, modifications to tools for compliance with governance standards, and a host of other issues. You may understand the mechanics of what was taught in the two-day course, but when you return to your own environment, you can't apply them.

The problems aren't limited to technical training. For example, when you learn product discovery practices in

a two-day training, you can grasp the practices, and you may even apply them to new ideas you have for your real-world products. However, the product ideas you work on in these short courses don't have the history of your own products—history filled with attachment to previous decisions, constraints of promises already made, and technical limitations.

More problems occur when you're the only member of your team at the training. You may learn the practices perfectly, but as soon as you return to the team, you quickly run into conflicts with people who haven't attended the training. You may be unable to apply what you've learned. For example, if you're the only one on your team who's attended a test-driven development course, it will be hard to get the full benefits of that practice if your coworkers are committing code without tests.

In our own experience with two-day training events—which, in all transparency, we teach in addition to the dojo model—we've seen that during the course, everyone's excited, understands the concepts, and knows how the new skills will be used when they return to their cubicles. After the training, however, people forget how to use those new skills, or the existing practices create obstacles to adopting the new practices. For this reason, we structure all of our training engagements to include exercises using real-world product development. We

include follow-up coaching, so we can address real-world constraints impossible to address in a two-day course.

CERTIFIED DOESN'T MEAN QUALIFIED

Standard training courses often focus on specific branded frameworks or commercial tools. This is so common in the Agile framework space we don't even need to mention the frameworks by name. Examples in the tool space include specific cloud platforms and DevOps tools. Attendees of these courses often receive certifications, sometimes advancing in levels by attending more courses.

Learning frameworks and tools has value, but selling courses and charging for certification tests has become a lucrative money-making industry in and of itself. Most importantly, having the certification doesn't always translate into having the skills the certification implies. Often there is no requirement to demonstrate competency. In other cases, certification requires passing a test; but how valid is a multiple-choice test in proving you know how to use a tool?

Consider driving: to get your driver's license, you don't just play a video game simulation. You have to drive in the real world. Even so, we all have a relative who is "certified" to drive a vehicle, but we wouldn't want to be in the car with them on a wintry night.

LEARNING ALONE IN A TEAM WORLD

Many organizations offer employees access to self-paced, individual training platforms. Employees can access books, video content, and online courses. These platforms have their place, but they're no substitute for whole-team learning. It does no good for a few people on a team to learn a new practice if that practice is not going to be supported by the whole team. In other cases, people spend time learning skills they find interesting, like new languages or frameworks, but those skills may not have a whole lot of value if the team is never going to adopt those languages or frameworks.

In addition, collaborative learning is more effective than individual learning. In *The Accelerated Learning Handbook: A Creative Guide to Designing and Delivering Faster, More Effective Training Programs*,¹ Dave Meier states, “All good learning has a social base. We often learn more by interacting with peers than we learn by any other means. Competition between learners slows learning. Cooperation among learners speeds it. A genuine learning community is always better for learning than a collection of isolated individuals.”

Our experience helping teams learn supports these assertions.

¹ Dave Meier, *The Accelerated Learning Handbook: A Creative Guide to Designing and Delivering Faster, More Effective Training Programs* (New York: McGraw-Hill, 2000), 9.

LEARNING TO RIDE A BIKE

We think we go to school—or a training course—to learn or gain knowledge. We tend to overemphasize metaphors like knowledge transfer, knowledge exchange, and knowledge management. We act as if knowledge can be moved around like any other commodity and that we will gain new knowledge through a simple act of consumption. The truth is, acquiring new knowledge is an active act of creation. Obviously, people with knowledge we don't have can assist us in the knowledge creation process, but they can't simply "give" their knowledge to us.

Remember when you learned to ride a bicycle? You probably saw a parent or older sibling, let's say your sister, on her bicycle, legs moving in circles, maybe laughing or whooping as the breeze cooled her face. You wanted to experience that freedom of movement. Your sister didn't give you a manual on riding a bicycle or stand in front of a dry-erase board and draw a diagram to explain the anatomy of your quads contracting to push your feet down to apply pressure to the pedal to turn the wheel, or the physics behind keeping your balance. She put you on the seat of a two-wheeler and pushed. At first the pedals pushed your feet more than your feet pushed the pedals, but then you got the hang of it. She ran alongside you, holding the back of the seat to help you understand balance. She may have done this many times. Eventually, she let go and you were pedaling and moving forward while the wind cooled

your face. She helped you create your own knowledge of how to ride a bike.

That's how humans learn.

There are two kinds of knowledge:

- Explicit knowledge can be written down (or captured in some other form, like video) and stored. When information is categorized, organized for consumption, and stored, it is being made explicit. Someone else can consume it without any direct contact with the person who documented the information. Hopefully, new knowledge will be created through the consumption of that explicit knowledge.
- Tacit knowledge, on the other hand, is knowledge we've acquired through experience and practice and is difficult or maybe even impossible to document. It is highly contextual. We may not even be 100 percent conscious of having it until someone asks us a question or a situation requires us to access it. Tacit knowledge shows up as intuition, gut feelings, and experience that tells you the decisions to make. Sharing tacit knowledge requires feedback loops, dialogue, answering questions, and validation that the person on the receiving end "gets it."

In the world of technology, there are a lot of things that

can be made explicit: we can share code examples, write tutorials, and document patterns for solutions to commonly recurring problems. A mistake that organizations often make is believing everything can be made explicit and that getting teams up to speed on the latest and greatest skills is simply a matter of giving people access to the right books or learning platforms, that is, the right forms of explicit knowledge. Organizations underestimate the amount of person-to-person contact needed to foster knowledge creation through the exchange of tacit knowledge. They also underestimate the amount of time needed for tacit knowledge exchange.

If this seems too theoretical or like too much of a straw man argument, ask yourself these questions:

- How many times has your organization sent one or two people to a training event or a conference with the idea that they would document what they learned and share that knowledge with the rest of their team and the organization?
- How effective has that been in helping the team adopt new skills or practices?

Organizations need to create holistic, immersive learning environments that balance explicit and tacit learning.

HOLISTIC LEARNING

The product development value stream comprises everything that happens during the product creation life cycle. It begins when someone recognizes an opportunity or a problem and has an idea for a product. The idea is refined during product discovery, gets developed, and is ultimately delivered to the customer in some form, where it provides value. The organization that creates the product captures analytical data about the product's usage, creating a feedback loop that flows into product discovery, so they can continue to improve the product and provide value to the customer. Clearly, a series of activities, events, and processes have to happen to take the product from idea to production, and there are various practices that address different segments of the value stream.

In many organizations, different people who aren't part of the same team work in isolation from each other. They work through different practices in different segments of the value stream. There is weak collaboration between them. Sometimes people working on building the product have no idea why they are working on particular features or capabilities. They are simply "coding to the spec." By the time we get to the people automating deployments with tools, we've lost all connection with the value a particular feature is supposed to provide to the customer.

What does this have to do with learning? A lot.

Training courses typically address small segments of the value stream. One short workshop might focus on product discovery, which is an early segment of the value stream; another might focus on one of the Agile frameworks, which typically address the delivery segment of the value stream.

You might send your domain experts and designers to a product discovery course, while you send your delivery team to an Agile framework course. Attendees of the product discovery course might learn new techniques for writing stories, like story mapping. People in the Agile framework course may be taught to write stories following the “as a (blank), I can (blank), so that (blank)” format. When the people who attended the product discovery course deliver their story maps to the delivery team, friction occurs. The delivery team may insist on rewriting the stories in the format they were taught. (We’ve seen this exact scenario play out multiple times.)

You’ve got two groups trying to learn how to do their jobs better and collaborate together, but they’re learning from different sources, different philosophies, and different value systems. They get bogged down in the mechanics and the differences in the way they’ve been taught as opposed to learning how to optimize the flow of work to deliver value to their customers. Ironically, the solution employed to address discrepancies like this in many cases

is to add more steps to the process, disconnecting people even further.

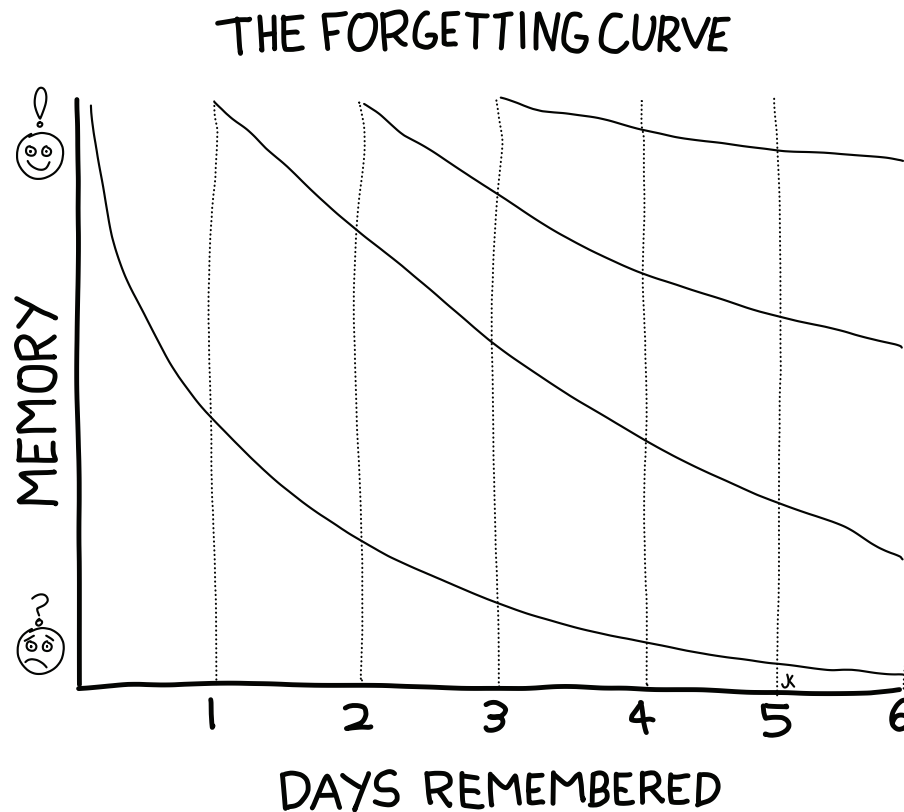
When teams come into the dojo, we teach them how to holistically address the entire value stream. The team within the dojo becomes larger because it encompasses the whole product development community and isn't limited to developers or the product discovery team. We aim to build overlapping perspectives and skillsets at multiple points on the value stream, so the entire product community begins to see how the pieces fit together to create products with better product/market fit.

LEARNING THAT STICKS

Ancient Greeks developed mnemonic devices to remember long series of words or numbers, and still today, people demonstrate superhuman memorization skills to earn a listing in the *Guinness World Records* book or so they can compete in World Memory Championships. But what about the rest of us? How does the average person learn and retain facts or skills, especially in a time when distractions and information overload are the norm rather than the exception?

Researchers have been trying to harness the secret to a better memory for centuries. In 1885, Hermann Ebbinghaus developed the Ebbinghaus forgetting curve theory,

which uses a mathematical formula to calculate how long we retain a fact or skill in relation to how much time has passed since we learned the fact or skill. The curve is steeper than we'd like it to be:



We forget more easily than we might think we do.

Modern research estimates we forget 90 percent of what we learn in a standard classroom setup in the first month after it's been learned—some even cite the first week. Ebbinghaus was way ahead of them. His research showed that the quality of our retention is based on two things: repetition (particularly spaced repetition) and the quality of the representation of the memory itself.

Learning in the dojo model is effective in part because it addresses both of these concerns.

Repetition is the name of the game in the dojo. Teams work in two-and-a-half-day sprints, which provide opportunities to work with new practices multiple times.

Representation of the memory itself refers to how you store the information in your mind. The quality of the memory representation improves when the memory is important to you and when you're able to create strong associations with the concepts you're learning. In the dojo, if you're learning skills in the context of the work that you have to do every day, you will create stronger associations than you would when learning sandbox exercises in a training course that has nothing to do with your daily work. The dojo creates an ideal environment for learning that sticks, that is, learning that people will remember.

CONTINUOUS LEARNING

A lot of organizations say learning is important. They budget time and funding for learning, but they also treat learning as a separate event that takes place outside the normal workday. The truth is, especially in IT, learning is given a lot of lip service and is only important when the team isn't busy—which is never.

When organizations build a dojo, they are making an investment in learning and acknowledging that learning can't be limited to one-off, separate events. When teams come into the dojo, the myth of learning being a separate event is quickly exposed. Teams discover that learning is an ongoing and continuous effort that happens throughout the entire value stream. They learn to acknowledge what they don't know and work on solving problems together, growing new skills along the way.

Teams leave the dojo knowing how to apply this continuous learning and improvement as they continue to work outside the dojo—they don't get that from traditional approaches to training, workshops, or courses.

TRADITIONAL TRAINING	DOJO
Practices are taught with predefined exercises completed, step-by-step, in clean sandbox environments.	Practices are taught while doing product development in the organization's environments subject to real-world constraints and limitations.
Learning is focused on the individual. There is a high risk of new skills not being adopted by the whole team.	Whole teams learn practices together, developing shared understanding and values. This supports long-term adoption of new practices and skills.
There is limited time for assessment, questions, and feedback required for tacit knowledge exchange.	Time for assessment, questions, and feedback is a core part of the dojo model. Conscious attention is paid to tacit knowledge exchange.
A single set of practices related to one skill is taught in isolation without providing context of how those practices fit into the bigger picture.	Practices related to multiple skills applying to the entire product development value stream are taught holistically. Learners understand how the practices are interrelated.
Exercises are done in a step-by-step progression with little to no repetition or practice. There is a high risk of quickly forgetting new information.	Repeated practice and application of new practices leads to long-term skill improvement, that is, learning that sticks.
Learning is a one-off, special event. Learning is viewed as being separate and distinct from daily work.	Learning and continuous improvement become part of a team's daily work. The organization becomes a learning organization.

Today, the complex products organizations develop require a high level of collaboration and a mindset of continuous learning. We can agree that learning is individual—you have to create new knowledge for yourself. In the context of individual learning, however, a collaborative space encourages sharing ideas and helping each other if we get stuck.

A dojo fills the need for a collaborative learning environ-

ment. A huge portion of working and learning in the dojo happens because we're supporting each other's learning and improving how we collaborate. We walk through an overview of the dojo experience in chapter 2 and then we'll unpack each part fully in the chapters that follow.

PRACTICES, SKILLS, AND OFFERINGS

We use the terms "practices," "skills," and "offerings" throughout this book. It's worth clarifying the meaning behind these terms, at least in the context of how we use them when we're talking about dojos.

Practices are specific techniques or processes teams learn to increase their skills. For example, test-driven development is a practice that helps teams increase their software design skills. Prototyping and customer interviews are practices that help teams increase their product discovery skills.

We think of offerings as the products your dojo offers. The "standard" offering we describe in this book is the six-week dojo experience where improvements to specific skills and related practices are defined on a team-by-team basis. Another offering we've delivered in dojos is a week-long design sprint focused on improving product discovery skills and related practices. As you consider creating offerings for your dojo, you'll want to consider what skills teams want to improve and what practices will help them improve those skills.

Chapter Two

THE DOJO EXPERIENCE

I have been impressed with the urgency of doing. Knowing is not enough; we must apply. Being willing is not enough; we must do.

—LEONARDO DA VINCI

The dojo is about learning by doing, first and foremost. Dojos don't have a standard curriculum. Teams work on gaining skills by learning and applying new practices to their real-world work. Whole teams learn new practices together and they learn them in the context of the constraints and limitations that exist within their organization.

Why do we do it this way?

LEARNING BY DOING

Teams often get bogged down in the decision-making process around what they should do next. There's a never-ending list of changes they could make to improve the product and ideas for technical improvements, whether the ideas are for cleaning up existing code or implementing entirely new technology. Often, teams become paralyzed while considering what to do next. When considering any change, they get bogged down thinking about all the possible ways things could go wrong before they begin to try different solutions. In the dojo, we encourage them to experiment and learn from doing the work.

One team in a dojo was struggling with deciding which way to implement an address lookup. There were plenty of third-party map tools they could leverage, but they didn't understand how each tool would affect the product. They didn't understand the technology. The team started going down a rabbit hole, discussing all the ways they could use the map tools. We said, "Let's just try one example with one tool, and then after we've done it, let's talk about what we've learned." The team spent an hour trying one of the map tools for the address lookup. Afterward, they discussed what they learned about the tool, the problem space, and how well the tool worked for the address lookup. The next time a problem came up, they started to spin around all of the possibilities for solving

it until somebody on the team said, “Why don’t we just pick an approach, try an example, and see what we learn.”

Another group was working on a product to keep store inventory up-to-date. There are many ways to update an inventory, and the group was, again, stuck in the decision-making process. Instead of researching the problem for weeks and going off on tangents, they were able to frame up what they wanted to learn about the problem. Then, they went through the learning together, shared what they learned with their leaders, and made a decision about how they were going to keep inventory up-to-date. The learning cycle was faster, because they intentionally chose to learn by doing. They owned their learning.

These events happen all the time while teams are building products, but seldom do they take the time to reflect upon what is happening, what is being learned. During a dojo experience, teams learn practices and skills and—just as importantly—they learn how to learn.

OVERVIEW OF THE DOJO FLOW

THE INTAKE PROCESS

The life cycle of a dojo experience begins with the intake process, which we fully describe in chapters 6 and 7. The dojo staff, who we introduce in chapter 5, conduct

the intake process, comprising the overview, consult, and chartering.

Overview

A person or a team will discover the dojo and be curious to learn what it is. They may walk past the dojo space and ask, “What is this place?” Or they might read about it in a company newsletter and seek out more information. Initial interest could come from a whole team, a single person on a team, or a leader higher up in the organization who leads multiple teams.

The overview is the first contact between the dojo and an interested party.

The first questions the dojo staff are asked are usually “What is a dojo?” and “How does it work?” The dojo staff answer these questions and start learning about the interested party. The conversation is light and informal, with the interested party getting a general understanding of the dojo and the dojo staff getting a general understanding of the interested party. The primary goal of the dojo staff is to steer the contact toward a consultation, or consult.

Consult

The purpose of the consult is to teach a team what a dojo is and to set expectations with a team around how the dojo works should they decide to sign up for a dojo experience. Team managers need to understand that the team will need time to learn. Managers need to make sure space is created in the team's schedule for the learning to happen. The dojo staff needs to understand what the team wants to learn, so they can help the team be successful. The goals of the consult are for the team members to understand the expectations of the dojo, to foster interest in learning together, and for the dojo staff to learn about the team's high-level learning goals.

The consult may be informal, or the dojo staff can prepare a presentation explaining what the dojo is and how it works. It's also a good idea to share experiences and results of teams who have been through the dojo. Sometimes, in the first consult leaders will want to have a conversation with the dojo staff without their team, and that's fine. It's important, however, that the whole team meets with the dojo staff in a second consult meeting. When the leader is ready to bring the whole team in for a consultation, it's a good sign that they're leaning toward committing to a dojo experience.

Assuming the team goes through the consult and decides to continue, the next step is chartering.

Chartering

Chartering (chapter 7) is a half-day activity scheduled to take place one to two weeks before the six-week dojo experience begins. Chartering can lead to the discovery of items that need to be addressed prior to a team starting in the dojo, which is why we charter at least a week before the team begins. The team meets with the coaches from the dojo staff to build a shared understanding of what success looks like for the team over the course of the six weeks. They establish a defined list of goals; if they hit all those goals, everyone would be really happy that they came to the dojo.

The team talks about what practices they want to learn and how learning those practices will help them. For example, a team could say, “By learning to automate our deployments, we expect to remove errors from the deployment process and spend less time troubleshooting why a deployment isn’t working as expected.” In chartering, we want to make sure we’re connecting learning goals with outcomes that will help teams deliver better products.

THE SIX-WEEK EXPERIENCE

As we’ve described, this book is focused primarily on the “standard” dojo offering, where teams go into a dojo for six weeks. A typical day involves learning tools and prac-

tices and applying them to product development. Teams work in two-and-a-half-day sprints. Chapter 8 outlines the cadence of the six-week experience.

Over the course of the dojo experience, the team will have twelve sprints in which they repeatedly practice what they learn. For example, a team might be new to TDD (test-driven development). The first test a team writes might be easy—they could choose a simple test without any dependencies on other systems. They understand the flow of writing the test, getting it to pass, and refactoring the code to make quality improvements. However, it's still shallow learning because it was just one test, and an easy one at that. Just because they understand the mechanics of writing the test doesn't mean they know how to apply those mechanics to a more complex scenario. The team writes another test; the product context is the same, but the way they need to test changes. Perhaps the first test was a happy path scenario with no external dependencies, and this next test has an error and one external dependency, like an alerting system. The concept of testing is repeated, but the approach changes. Deep mastery occurs through repeating the practice in different scenarios, learning how to make adjustments along the way. The multiple small sprints provide opportunities for practice and repetition, fostering deeper learning and stickiness—the retention of that learning.

Smaller sprints also encourage teams to reflect on how they are learning together, something they often neglect. Although retrospectives are a fundamental aspect of Agile methodologies, reflecting on learning is often a new concept for teams. The idea is to reflect on how well the team is meeting its learning goals and to see if the team would like to try anything different. Teams pause at the end of a sprint and ask, “What would we like to try differently to foster learning during the next sprint? What do we expect will happen if we make this change?” We ask the team to think of ways that they would know the change was a good idea—to give an example of how they would measure the impact of the change, even if it’s subjectively. In this way, the dojo is fostering intentional learning and continuous improvement. Repeating the retrospective frequently within the dojo makes it a habit the team takes back to their normal work environment.

Learning in a dojo happens through experience. Teams revisit their goals often during the six weeks. They think about whether they’re learning the right things and whether or not what they’re learning is helping them make a better product.

POST-DOJO

When the team concludes the six-week experience, the dojo staff conducts exit interviews to assess how well

the experience went for the team. Based on those interviews, they develop a transition plan to ensure that the new practices, the new style of working, and the focus on continuous learning and improvement continue when the team returns to their normal work environment.

Chapter 9 fully discusses post-dojō activities.

WHAT MAKES A DOJO A DOJO?

Dojos are gaining enough notoriety that companies are starting to rebrand existing products as dojos. Products that were previously called “labs” now have the word “dojo” in their name. What looks like standard Agile coaching that’s been around for years is now being called a dojo. More and more consulting firms are offering dojo-related services. We think this is great—assuming that what’s being called a dojo is indeed a dojo. Unfortunately, in some cases it may just be an attempt to cash in on a good idea.

(NOTE: we are not referring to the use of the term “coding dojos” here. Coding dojos have been around far longer than the dojo model we describe in this book. The two share common values around deliberate practice, group learning, and repetition when learning new techniques and practices. For those interested in learning more about coding dojos, Emily Bache’s book *The Coding Dojo Handbook* is a great reference.)

We are often asked “What makes a dojo a dojo?” Dojos are pragmatic by design. Organizations create dojos to meet the needs of their teams where they are at and to teach teams skills that will help them deliver better products in their specific markets. This may result in the creation of dojos that vary significantly from organization to organization.

Here is what we see as being negotiable and nonnegotiable principles for calling something a dojo.

Nonnegotiable

- Learning over delivery
- Whole-team learning
- Spaced repetition and opportunity for repeated practice
- Skilled coaching
- Learning in context of real-world work
- Collaborative peer-to-peer learning
- Safe-to-fail learning (in small increments)

Negotiable (assuming nonnegotiable principles are met)

- Offerings
- Duration of offerings
- Skills and practices learned
- The physical space

When starting your dojo, we suggest you start with the six-

week offering. You can add additional offerings once you've established the six-week offering. However, providing too many offerings of differing durations could be confusing for potential teams.

Over time, your dojo will become established as a known product within your organization. You will learn from your experiences with teams. You may discover there's an opportunity to address specific skills or problems with offerings of shorter durations. Some dojos have created a one-week offering focusing heavily on product discovery with user research and prototyping. Other dojos have created short two- or three-day experiences where teams can come in to learn a specific technology, like Kubernetes. Another dojo even created an extended offering lasting ten weeks where people brand-new to technology learned how to write code and build products! All great ideas...but not right off the bat. Early on, make it easy for your customers to say "yes" to your product by limiting your offerings.

A SAFE SPACE TO LEARN

Going back to the Ebbinghaus curve—without the opportunity for practice and repetition, learning will not stick. Often, skill growth is just expected to happen while people are doing their work. There's no space in the schedule for the learning to occur. And there's absolutely no room for failure—even on a small scale and even though we often

learn more from our failures than from our successes. In most organizations, teams simply can't ever be wrong.

The ability to innovate is lost when there's no room for failure or learning. People become conservative, falling back to what they already know instead of trying to apply new ideas. Or, when they have to implement new technologies, they'll do it without really understanding them. Asking people to adopt new practices and acquire new skills without giving them time to learn those new practices and skills results in poor quality products. By its nature, a dojo becomes a space where applying new practices and skills has a lower emotional and financial risk. In the following chapters, we explain how to choose the practices you'll offer, how to set up the space, and who will run your dojo. We then walk you through the steps for running your own dojo experience.