About the Author

Katherine Radeka has a rare combination of business acumen, scientific depth and ability to untangle the organizational knots to remove the barriers to change. Since 2005, Whittier Consulting Group, Inc. has helped some of the world’s leading companies get their products to market faster. She has a global reach with clients in Europe, North and South America, Asia, and Australia/New Zealand. She has worked with companies in pharma, biotech, medical device, high tech, consumer electronics, food and beverage, and consumer packaged goods, among others. She currently supports over 50 implementations of Rapid Learning Cycles through the Rapid Learning Cycles Certified™ Professionals Community.


Katherine’s second book is *The Shortest Distance Between You and Your New Product: How Innovators Use Rapid Learning Cycles to Get Their Best Ideas to Market Faster*. This book summarizes Katherine's ground-breaking work to integrate Agile Development with her work on Knowledge Capitalization into a proven method for accelerating innovation.

Katherine has climbed seven of the tallest peaks in the Cascade Mountains and spent ten days alone on the Pacific Crest Trail until an encounter with a bear convinced her that she needed a change in strategic direction.
Design-Experiment-Capture
Close Knowledge Gaps Scientifically and Permanently

Key Takeaways

• Build-Test-Fix mode is our default mode for product development - it's the most intuitive process even though it does not lead to good results and takes a long time.
• Design-Experiment-Capture summarizes the process for closing a Knowledge Gap scientifically and permanently.
• LAMDA and other problem-solving methods based on the Scientific Method can enrich this process to build better experimental designs.

You may have noticed that in the past couple of months, we've attempted to launch a completely new email template for our Knowledge Brief of the Week emails that is much more image-driven, and gets me out of the weekly publishing process. It took me much longer than I anticipated to get the first one out, because it was so different from what we'd done in the past, and there were a lot of new steps to prepare the graphical elements. Without realizing it, I had a lot of Knowledge Gaps to close about the best workflow for preparing the right images in the right sizes, for testing the responsive design across multiple browsers, mobile phones and tablets, and for how to set up the template so that someone else could put together the template once I had done the writing.

I wish I could say that we defined all of these Knowledge Gaps and used Rapid Learning Cycles (or even LAMDA) to close them - but alas, no - the task seemed too small to bother with all of that. Therefore, I went into default mode and this project devolved into several days of Build-Test-Fix loops. It only got better when I realized what was happening, and then set out to close the specific gaps that were causing me the most trouble - and finally admitted that I needed to revert to something easier. It's taken us some time to solve this, but we have now solved it permanently.

When we're in the midst of development, especially if we are hot on the trail of a new idea, we can easily lose sight of the rhythm of the entire process in the search for solutions to individual development challenges. Even if the team is using Agile program management methods for frequent check-ins, they can still lose sight of the big picture if they are focused on their activities instead of focusing on what they need to learn. This is why we emphasize the need to close Knowledge Gaps systematically and permanently - otherwise, product development devolves into a series of Build-Test-Fix loops because it's the most intuitive way to work - but our intuition is wrong.

Build-Test-Fix is the Default Mode for Product Development

Build-Test-Fix is the mode that we fall into whenever we have an idea that seems easy to realize. We try something, test it, and then iterate on it, trying lots of different things until something finally works. The solutions to our problems seem just out of reach - surely we'll be able to fix them with just one or two more tweaks. We can be more or less systematic about this - thinking the problem through analytically, or just throwing a lot of ideas into the mix to see what sticks. Rarely are we able to remember afterwards the path we took from A to Z, and we could never write down what happened. We also had no idea how long it would take to reach a viable idea. We might get lucky and hit it right the first time, or we may be there through the night, loath to go home because we are "so close" but ultimately forced to give up in defeat if we can't find something before we burn out.

Product development experts have known for a long time that Build-Test-Fix only looks productive. In The Lean Startup, Eric Ries attempted to improve on this with Build-Measure-Learn, and for a team working in the Software/IT world where it's feasible to build shippable features in one cycle, this is a big improvement. But Build-Measure-Learn isn't a good approach for products that are not so easily changed, and not so easy to break down into "shippable" features. For products that have to obey the laws of physics, chemistry and biology, building a product may not be the best way to run an experiment that will build the knowledge you need to close Knowledge Gaps. The best experiment to evaluate a new manufacturing method may look nothing like a final product.

From Build-Test-Fix to Design-Experiment-Capture

A mechanical engineer or a chemist needs a different type of cycle to close a Knowledge Gap. Design-Experiment-Capture. It takes a good plan to run a good experiment, and then the data from that experiment doesn't become knowledge until it's captured. Learning is not an explicit step in this cycle - it happens throughout the Learning Cycle.

Step 1: Design the experiment

Good experiments require some experimental design to ensure that the data from the experiments will contribute to closing the Knowledge Gap. An experiment that generates inconclusive data still generates knowledge, but not the kind of knowledge that will lead to a better Key Decision. Experiments are more likely to generate useful knowledge the first time if you have taken time to think through the hypothesis you'll test, and then design the experiment and the data analysis methods to test that hypothesis. This is also a good time to make sure that you understand the approaches others have taken to work through similar Knowledge Gaps.

Step 2: Run the experiment and analyze the data.

Whether the scientist runs the actual experiment or designates the tasks to lab technicians, it's important for someone to write down their observations as the experiment is running. While the final results will come from data analysis, these observations can provide additional context that brings the data to life for the people who have to make the final decisions. We capture these on the Knowledge Gap report under the "What Have We Learned" right next to the charts that show the results of our data analysis.

Step 3: Capture the knowledge

A Knowledge Gap is not closed until the Knowledge Gap Report has been written and accepted by the team. We intentionally designed these reports to be relatively quick to write, and to include only the most important information. An experimenter may write a formal lab report, but that report will probably have many fewer readers, since only those who need to know the details of the experiment will bother to open it. For most people, including team members in related subteams and other stakeholders, the Knowledge Gap report will have sufficient detail to support making better Key Decisions, and for evaluating whether the knowledge is reusable to help a future team adopt a Known Solution from this product or close a related Knowledge Gap of their own.

What About LAMDA?

For many years, I've advocated for LAMDA as a means to close Knowledge Gaps, and I still think that leads to better experiments. The LOOK and ASK steps encourage the researcher to get a good understanding of the current state of the problem and the current state of the research. MODEL is the Design-Experiment-Capture process outlined here for developing and running the actual experiment. DISCUSS is what happens at the Learning Cycle and Integration Events, and ACT is the implementation process for incorporating the decision into the design.

The problem that I've encountered is that LAMDA needs to be taught first and then someone needs to provide mentoring for a person to use it well. It's not a natural fit to replace Build-Test-Fix unless the organization is already using LAMDA in other contexts. I don't have the time to teach LAMDA alongside Rapid Learning Cycles in the public workshops, which are more about how to establish and run a Rapid Learning Cycle. We start by breaking the Big Problem down into a framework at a program level - LAMDA takes place at a level underneath that.

Most modern problem-solving methods (LAMDA, 8D, Seven Step Problem Solving, PDCA and even Eric Ries Build-Measure-Learn) have their roots in the Scientific Method that many of us learned in grade school: they all ask the researcher to develop a hypothesis, design a test, run it, analyze, reflect and capture the results. Over time, I've learned that we need to keep things as simple as possible if we want to overcome the temptation to move into default 'Build-Test-Fix' mode.

Close Knowledge Gaps Scientifically and Permanently on Every Project

Every product development project - even software and IT programs - could benefit from a scientific approach to identifying and closing Knowledge Gaps. If we already knew everything we needed to know to deliver a new product, then we wouldn't need a development project in the first place. On a small project, it may only take a team fifteen minutes to uncover a handful of Knowledge Gaps that would benefit from some targeted learning to avoid getting stuck in endless Build-Test-Fix loops. The fallacy that I fell into around the email template was that this was too small to establish the infrastructure for good decision-making. So I didn't reflect on the fact that my staff might need new Photoshop and Dreamweaver skills to make the template work. Once I did that reflection, it became a lot easier to see what I knew - and what I still need to know - in order to deliver a Knowledge Brief of the Week every week. It turned out that we needed a few Rapid Learning Cycles to ensure that our new email process continues to get better at delivering relevant content and helping us spread the word about Rapid Learning Cycles.