Agilent Case Study

Kevin Thompson, Ph.D.

www.kevinthompsonphd.com

March 5, 2016

Contents

1 Case Study: Going Agile on a Grand Scale 2
2 More complicated than mobile apps 2
3 A half-hearted Agile attempt 3
4 Learning to stack marshmallows iteratively 5
5 Getting on the same page 6
6 Starting at Sprint Zero 6
7 Transformation side effects 7
8 Goal: Baked in debugging 8

Abstract. This case study describes the Agile transformation of Agilent Technologies' Software & Informatics Division, a globally distributed organization of more than 100 software engineers. Facing a history of poor release predictability, with fewer than 20 percent of releases delivered on schedule, the Division undertook an organization-wide transformation involving Scrum training, coaching, standardized practices, program-level coordination, and integrated work-management tools. The transformation improved transparency, estimation, cross-team coordination, and organizational alignment. In August 2016, the Division delivered its next-generation chromatography data system software on schedule—a milestone unprecedented in its recent history.

1 Case Study: Going Agile on a Grand Scale

In early 2015, Agilent's Software & Informatics Division set a new priority: Improve the predictability of the group's performance.

The need was clear. At one meeting to review the readiness of software two months in advance of its release, the team discovered 185 points in a backlog that had not been put into the scope of work. The product, already more than a year in development, was delayed another 4 months. The case wasn't an anomaly. The organization was meeting fewer than 20 percent of its release deadlines.

"We needed to change significantly the way various functional silos coordinated to build products," says John Sadler, VP and General Manager of the division. He proposed an Agile transformation and in August 2015 greenlighted a coaching relationship with cPrime that included a JIRA project management software integration. One year later, when the division released next-generation software to support Agilent's chromatography data systems, it was the first time anyone in the organization could remember delivering on schedule.

2 More complicated than mobile apps

Getting there was no small task. Agilent's Software & Informatics Division includes 150 employees on two continents who collaborate with contractors on a third. Comprised of engineering, marketing, quality assurance, tech support, and sales professionals, the Division develops the tools that power Agilent hardware—state-of-the-art analytical instruments used in industries including pharmaceuticals, chemicals, energy, plastics, food, and environmental remediation.

Agilent's customers in those regulated markets are required to validate each step of their process. They put new equipment into production over lengthy cycles, and carefully schedule downtime for software upgrades. Stefan Weiss, Integrating Manager at Agilent-Technologies in Waldbronn, Germany, explains, "We can't update our software on a daily basis like mobile app companies can without the customer noticing. Our customers don't like updates, they don't like bugs, and the cost of validation is multiple the cost of an update."

Indeed, the constraints of developing software for scientific analysis instrumentation would seem to contradict some of the principles of Agile development: Agilent ships machines on three-to-five year cycles to customers who expect to use them indefinitely; the company's installed base is expansive, including some customers who maintain offline labs without access to downloads stored on the cloud; and because Agilent develops and delivers software on multiple continents, daily standups—a mainstay of the Agile method—are impossible.

Still, an Agile approach was the best hope for improving predictability and transparency, as well as Agilent's ability to win and retain business. "In order to mediate among all of the interests we serve, we had to be able to forecast with a reasonable degree of confidence how long it was going to take us to get things done," says Sadler.

He proposed to his team an adapted version of the Agile Manifesto, with caveats:

- Individual interactions beat processes and tools *when individuals truly interact across functions on a daily basis (we can do this)*
- Working software is better than comprehensive documentation *when the software is simple and non-technical (challenging for us)*
- Customer collaboration wins over contract negotiation *when customers are available for frequent collaboration and we can tell objectively what works for them (we can use proxies, but must bear in mind that deployment cycles are long)*
- Responding to change beats following a plan *when we have frequent customer interaction, objective feedback, and the ability to deploy on short cycles (remember those three-year instrument cycles?)*

3 A half-hearted Agile attempt

To be sure, parts of Agilent's Software & Informatics Division had attempted to adopt Agile methods years earlier. But the effort had been less than thorough and inconsistent across the organization's six collaborating software teams in Santa Clara, Waldbronn, and Grenoble, France. Several team members now acknowledge that, while they claimed they were doing Agile, the six teams used two incompatible work-tracking systems and each followed a different "Agile" approach. Agilent Software Project Manager Rajesh Parikh explains it this way: "We were using Agile, but not everyone was thinking Agile."

To support a process overhaul, David Barber joined as Senior Director of R&D for Software & Informatics. A seasoned engineer whose qualifications included experience working with Agile pioneers Mike Cohn and Ken Schwaber, Barber was the outsider the company needed to rally a mindset shift.

Barber, in turn, chose cPrime to train and coach the Software & Informatics Division through a true Agile transformation. Of cPrime Agile Practice Lead Kevin Thompson, Barber says, "We're aligned. There are different approaches to Agile, and Kevin has a pragmatic approach that says, 'This is what works and this is how we do it.'"

"We needed to engage with someone who could provide that common level of training, understand our complexities, and be willing to take us on from an established company with enough rigor to train worldwide teams," adds Agilent Project Manager Sonja Cuffe. "One thing we appreciated about cPrime and about Kevin in particular is that his thoughts and processes and approach to things doesn't waiver. You can't be too wishy washy in terms of describing the process when you're dealing with a group of intelligent developers. A consultant should come in and say, 'Hey, this is how we think it should be.' It has been very beneficial having Kevin on board."

cPrime's JIRA expertise was also attractive to Agilent. The hardware maker had already decided to adopt the Atlassian project management tool, and would require a consultant to customize it. cPrime provided one-stop shopping. "JIRA offers by far the lowest initial cost of ownership among Agile product management tools, but it's hard to configure and not useful straight out of the box," Thompson says. "Agilent has people in California, Delaware, Germany, and France who need access to the same plans and information. They need a tool that can span the globe. JIRA is something they couldn't do without, and we did the installation and training."

A year after starting their Agile transformation process, Agilent team members describe various "before" and "after" scenarios that illustrate the dramatic change cPrime's approach enabled:

> Before: Performance might be measured in engineering hours, or person days, or Story points.
> After: Everyone agrees on how to measure Velocity and Burndown.
>
> Before: Teams with different levels of Agile training had different definitions for Epics, Stories, and Subtasks.
> After: Everyone in the organization speaks the same language.
>
> Before: Scrum masters wrote code, led Team meetings, and weighed in on feature priorities.
> After: Scrum masters are task masters who report up to product managers.
>
> Before: Features with bugs could get approved, carrying defects into subsequent development phases.
> After: A universal definition of “done” ensures that bugs are eradicated before the next Sprint.
>
> Before: Problems often popped up at the final hour, causing panic and substantial delays.
> After: A shared set of tools used across all Teams prevents surprises and helps maintain focus.
>
> Before: There was no instrument for measuring a Team's capacity to do work.

After: There is agreement on how to predict and measure capacity on a daily basis.

Before: Product teams burdened engineers with poorly written requirements.
After: Engineers do not accept unclear requirements.

Before: Sprints were derailed and engineers were whipsawed whenever marketing got new information.
After: Marketing shares new information when work is reviewed biweekly.

Before: No Team had ever completed a Sprint without interruption.
After: In August 2016, the Division delivered software with the prescribed feature set as scheduled.

4 Learning to stack marshmallows iteratively

How did this group pull off a successful Agile transformation on such a broad scale? The insiders credit some complex global logistical coordination, top-down and organization-wide buy-in, and cPrime's approach for large-scale Agile Governance.

Thompson documented that approach in a 2013 paper called *Recipes for Agile Governance in the Enterprise* (RAGE). Agile Program Management according to the RAGE model relies on key Ceremonies such as the Release Planning meeting, Team Scrum-of-Scrums meetings, Product Owner Scrum-of-Scrums meetings, Release Backlog Grooming meetings, Release Review, and Release Retrospective meetings. It also defines Area Product Owner roles and Program Manager roles, and measures progress with Burn-Up charts.

Thompson guided Agilent through the construction of a large-scale Agile framework by helping the organization to adopt a minimal set of lightweight decision-making techniques, execute Ceremonies, establish Artifacts, and learn other Agile practices.

Simultaneously, he helped the organization identify the necessary roles and individuals to fill them and guided a restructure that integrated development and operations. "Area Product Owner and Program Manager roles are critical for maintaining alignment between business goals and Team-level development," Thompson says. He spent five days at Agilent providing guidance on setting up a Program-Level structure, roles, and practices.

"An Area Product Owner owns the big-picture product definition, decides when to make major tradeoffs between functionality and schedule, and meets weekly with up to five Team-level Product Owners," he explains. "A Program Manager, meanwhile, facilitates most of the Program-level Ceremonies, monitors and manages cross-Team Dependencies, and generally strives for effective execution across all Teams. That's important, otherwise you have 14 Teams moving off in random directions and you don't achieve your goals."

To improve communications and enable collaboration, cPrime also migrated the entire Software & Informatics Division onto a single suite of work-tracking and collaboration tools from Atlassian.

The Division-wide transformation officially kicked off in September 2015. With its "Agile for Teams" classes in Santa Clara and Waldbronn, cPrime established a common vocabulary and singular understanding of Scrum concepts. All six Teams—more than 100 software engineers—plus Scrum Masters, Team Product Owners, Area Product Owners, Program Managers and their Managers attended.

Agilent Technical Marketing Manager Shawn Anderson describes how participants warmed up to Agile thinking with the Marshmallow Challenge: "All 100 people split up into small groups and competed to erect, from spaghetti sticks and string, the tallest structure that could hold a marshmallow." What did they learn? "The marshmallow is heavy enough to make the structure collapse, so you need to be iterative and put the marshmallow on at each stage before building your next level," Anderson says. Waiting until your structure is finished to put the marshmallow on is like the Waterfall method of development, he says, in which "marketing writes long instructions, you develop the product, and six months later marketing looks at it and says, 'I didn't ask for this.'"

5 Getting on the same page

In addition to the Teams classes, in late October, Thompson led all Scrum Masters, Team Product Owners, Program Managers, Area Product Owners, and their managers in a one-day Agile Program Management class. Some attended the early-morning Santa Clara-based course in person, while others joined via video conference from Waldbronn. To ensure top-down buy-in for the new approach, all Division leaders participated in a full-day Agile for Executives briefing. Thompson also coached Agilent's three Santa Clara software Teams in Story-writing, Backlog grooming, and Sprint planning.

Simultaneous to training Agilent staff in Agile process, the JIRA setup was underway. Moving all of Agilent's developers in Europe and the US to a single shared platform—as well as migrating Agilent's legacy data from Version 1 software to JIRA—would enable global access to plans, requirements, and work status across all Teams.

Stefan Weiss, who was responsible for reengineering the work tools, says Agilent had already introduced Atlassian's Confluence collaboration tool and wanted to implement JIRA's Agile project management tool, but needed help making the tool useful to its global team. cPrime's JIRA expertise made it the perfect partner for coaching the transition. It trained the global interdependent Teams to rely on JIRA as an organized place to see what work was going on within a Team and across Teams, when it would be done, and which features were on schedule for the current release.

6 Starting at Sprint Zero

By November 2015, 14 Agilent Teams were ready to get to work on Sprint Zero—a two-week planning session. Its outcome would be an eight-month product release plan for the Software & Informatics Division's most complex current project—software for Agilent's OpenLAB chromatography data system.

During the first week, Teams in Germany, France, and California worked in parallel. Another Agile coach, Bruno Orsier, R&D Team Manager from Persistent Systems, led the Teams in Agilent's Waldbronn and Grenoble offices, while Thompson coached in Santa Clara and provided an overall organizational model

and transformation structure. For the second week, all Team members and other relevant staff gathered in Santa Clara to finalize the plan.

Thompson and Orsier broke Sprint Zero activities into three categories:

1. Educate everyone on the business and technical goals, drivers, and high-level requirements for OpenLAB through presentations and posters.
2. Use the newly learned techniques to develop requirements and estimate Stories, Epics, and Defects.
3. Lay out Stories, Epics, and Defects on the Release Planning board, and mark all cross-Team Dependencies.

By the end of the second week, the Teams' eight-month development plan, laid out on a giant Scrum Board, dominated an Agilent auditorium. Chevron-patterned ribbon crisscrossed an entire wall, connecting the blocks on a grid of blue masking tape. The Plan comprised 14 horizontal lanes—one per Team—and 15 vertical Sprint columns. Teams noted their Sprint Velocities for each Sprint before filling the board with Stories, Epics, and Defects. Explains Anderson, "cPrime led us through sessions where one Team would explain what they were doing, and people listening could say, 'Oh, wait, you need this bit of information from our side to do that, so they would note a Dependency."

Thompson held the group to several rules:

1. The amount of work done in a Sprint cannot exceed the Sprint's forecasted Velocity.
2. Epics—high-level summaries of major deliverables—may cross Sprint boundaries as placeholders, but Stories and Defects may not. Every Story or Defect must be completed within the same Sprint in which it begins.
3. Every cross-Team Dependency must be denoted with the chevron-patterned ribbon pointing in the direction of Dependencies, from the predecessor Story to the successor Story.

By the end of the week, the Teams had successfully finalized a Release plan and recorded all details in JIRA. And though there were numerous "small problems and teething pains," says Thompson, they were "the sort one would expect in new Teams, not show-stoppers."

Through their next few Sprints, Thompson mentored the Santa Clara Teams while Orsier mentored the Waldbronn and Grenoble Teams.

7 Transformation side effects

Nine months later, members of the organization on both sides of the Atlantic Ocean use terms including "phenomenal," "dramatic," and "super successful" to describe their Agile transformation. Leaders are universally elated by improvements in predictability and transparency, and others inside the organization say they are happier and that overall morale has improved.

"We've become more disciplined at doing estimates, and the process of estimating work is getting better, so we're becoming more and more accurate," says Anderson. "Morale is definitely better. We

schedule vacations and holidays into release plans—the natural process of Agile provides breaks as well—and everyone is happy that we're meeting deadlines."

Parikh says adjusted expectations throughout the organization have improved life for software developers as well as customers. "cPrime met with people at different levels in all aspects of our unit, so now everyone has the exact same expectation and can set the right expectation for the customer," he says. "Everybody knows what is expected from them, and that makes people happy."

It's not that getting everyone on the same page improves the product, Parikh says, but because it improves the process, the product is enhanced.

Barber agrees: "Not only are the engineers happier, but the product teams and general management are too because products are coming out on time, we're hitting our checkpoints, and transparency is such that when there is a problem people start working on it sooner rather than delay the release. Stuff is getting done and it's on time. Who wouldn't be happier?"

8 Goal: Baked in debugging

What's next? Continued progress, Agilent executives say. This year, Thompson and Orsier began training and coaching the organization's Core Product Teams on the same Agile transition process. And the software group will improve even further.

"We'll continue working on the next generation of software this way," Barber says. "It's not a switch that you flip where one day you're doing Waterfall and the next you're doing Agile. You work it in gradually. The introduction to Scrum is done. Now we'll be improving our efficiency and we'll be getting better without working on hardened Sprints."

For instance, he explains that a lot of time for fixing bugs was "baked in" to the first release plan, with as much as 30 percent of the release time reserved for hardening. "We're going to try to squeeze that out. A well-operating team produces fewer bugs," Barber says.

He expects to see several more releases before the Teams are operating at top speed. Then, he predicts, "there will be an evangelism and other groups within Agilent will look to see what's going on."

Sadler praises the group for its progress. "For the first time in this Division's recent history, we're about to deliver a high-quality product with the feature set we said we would, on time, on a fixed release cycle cadence that allows us to plan out a good three years in advance with reasonable credibility. That's dramatic."

Still, he says, he's not about to declare victory. His organization, he says, is going to be relentless at hunting down waste: "The goal is not to be perfect. The goal is to be better."

For cPrime, however, its *Recipe for Agile Governance in Enterprise* is getting closer to perfection.