

If you're not a software developer, you might wonder why you should care what methodology you follow to build your software. We think methodology is critically important. And here's why...

- **Iterative Development:** We strongly believe that the while the core of an idea can be solid, in most cases the first implementation of the idea in code will be flawed. Agile software development favors shorter development cycles with multiple iterations. User experience and interface design benefit immensely from many successive iterations that build on previous lessons learned from previous attempts.
- **Reduce the Feedback Cycle:** The traditional method of software development that relies on user acceptance testing to provide feedback to the development team causes delays of weeks to months in the completion of a project. With a shorter feedback cycle, requirements or design defects are found and addressed very quickly and the project faces much less risk due to misunderstanding or mistakes.
- **Regular and Required Client Interaction:** The Agile process requires that the client or product owner is directly involved with the planning and execution of the development process. In some instances, the product owner is a full-time member of the team. This constant interaction helps to solidify the goals of the project and enables the team to adapt quickly to any changing priorities.
- **Testing from the Beginning:** We think that great software starts with great testing – not as an activity, but as part of the culture. Everyone on the team should be thinking about the best and more effective way to test. We are strong advocates of test-driven development, continuous integration, pair programming, exploratory testing, and effective test automation. For us, testing is part of development in every way – it's not a separate activity and testers are core members of the development team.

"Michael has been one of the best investments Weblink has made in the past 10 years; he has helped us accelerate our product development efforts by helping to build a world class development team. I would highly recommend Michael (except to our competitors) to any company considering his services."

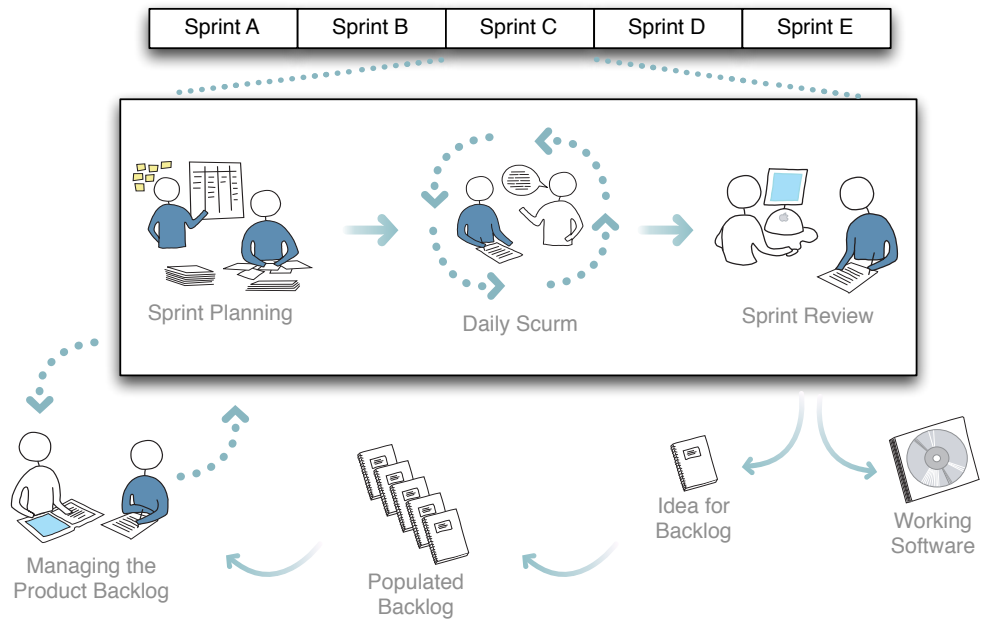
-DJ Muller, President/CEO at Weblink International

Scrum

The Developer Town methodology is rooted in Scrum. Scrum is an agile development methodology with special emphasis on transparency and accountability. It focuses on keeping the client and the development team in sync throughout the development of the product. In contrast to traditional development methodologies, which involve a time-consuming and error-prone requirements gathering processes before the project can begin, Scrum leverages a *Product Backlog*. The Product Backlog is populated with bite-sized features (or *Stories*) that can be implemented within a small, defined time period called a *Sprint*. The size of a Story is measured in *Story Points*.

Once the product backlog is populated, we hold a *Sprint Planning* meeting. In this meeting, we work with you to prioritize what work will get done in this Sprint. For a given Sprint, the team will use the Story Point measure to determine how much work can be pulled into the Sprint. For example, if the team has capacity to take on 8 Story Points this Sprint, you can pull in 4 two-point Stories, or 8 one-point stories. Since we charge by the Story Point, you can think of this as a short fixed-bid window of effort. Together, we agree what's in scope, and how many total points the team will deliver on.

At the end of the sprint, we hold a Sprint Review meeting where everyone gets back together to demonstrate what was built during that Sprint. The goal is to show working software in action, but many times we will also show off documentation, tests, and other artifacts in addition to the software developed that Sprint. After the demo, the Product Owner is responsible for 'Accepting' the story as complete. If the story is not accepted, the team either continues to work the story until it's accepted, or the story (or a revised version of it) goes back on the Product Backlog to be pulled back into a future sprint.



How does agile coaching work?

Agile coaching is about coaching a team – not an individual. While individual coaching may occur from time to time, the focus is on how the team and organization interact to produce great software. In our coaching engagements, we focus on:

- **Planning:** Effective planning doesn't start at the sprint-planning meeting. It starts before that with a robust well-ordered backlog, a product and technical roadmap, and with an understanding of core short-term and long-term goals. We work with teams to help them develop techniques for short-term and long-term planning, building great backlogs that teams can manage and estimate, and tips for running effective planning meetings that don't get derailed by design discussions.
- **Estimation:** Software estimation is difficult to do. Even the best developers struggle to provide accurate estimates for problems that they are solving for the first time. We work with teams to develop techniques to reduce estimation risk, produce more accurate estimates that anyone on the team can deliver against, and to provide visibility and consistency in delivery.
- **Review:** It's easy for an otherwise great sprint to leave everyone with a bad taste in his or her mouth because of a poor sprint review. From running the meeting, to making sure code is delivered and staged for review in advance, we can help teams deliver reviews that make product owners say "Wow." We also help teams develop techniques for effectively capturing the priceless feedback that occurs during review meetings, and for dealing with disagreements around story scope and acceptance.
- **Retrospectives:** The process that works today, won't work tomorrow. The only way you can be sure you're tuning your process to be it's most effective is through team retrospectives. These aren't just one-hour meetings where everyone talks and no change happens. These are focused facilitated sessions with specific deliverables and action items for the *very next* sprint. The goal is to keep energy high, keep people engaged, and to keep delivering better software.
- **Daily Scrums:** While it's a small thing, the daily scrum is the glue that holds a sprint together. Running these meetings effectively is more than just going around the circle for 15 minutes and listening to people talk. You need to make sure the team is actually *communicating*. This includes seeding the conversation with possible issues, looking at burn-down charts, and figuring out and following up on blockers.

- **Test-Driven Development and Continuous Integration:** Agile teams need feedback. Even if you're not doing TDD, you need some level of build automation and testing, and you need to get feedback when each developer checks in changes. That means you need to have CI running and you need tests executing against the code. From basic setup to looking at code coverage, checkstyles, and build trends, we can help your team become driven by the build process. When done correctly, it quickly becomes the heartbeat of the project.
- **Pairing and Code Reviews:** Silos destroy agility. To that end, we teach practices that help distribute responsibility and expertise. In addition, we also are big believers in pair programming and regular code reviews. Not only do these practices deliver better code – they deliver better developers. We teach teams how to include these practices in their estimates, how to manage them effectively, and how to be selective to make their usage most effective.
- **Dealing with Emergent Architecture and Documentation:** A big challenge of moving so quickly is dealing with emergent design and documentation. Teams need to develop a balance between comprehensive documentation and no documentation – and that's hard. We help teams identify which artifacts provide long lasting value, and we share techniques for how to work documentation and design into your stories to make sure they aren't being overlooked until it's too late.
- **Tools:** Finally, all of these activities take place in an environment full of tools that help with all tasks - from the critical to the mundane. We believe the right tools can amplify our effectiveness, while the wrong tools can drag down both energy and productivity. We also believe the "right" and "wrong" tools depend greatly on the team doing the work and the product being developed. We don't claim to know all the "best" tools out there, but we can help you figure out which tools are working for you and which ones aren't.

"Mike was hired to bring structure and best practices to our team to improve how we develop software. His coaching helped pave the way for a new company launch and a much better foundation from which to scale our future development in existing businesses. I expect Mike to be involved as a coach to our business any time we know we need to get better."

-Scott Hill, CEO at C.I.K. Enterprises

"[Mike] does not inflict his help on you, he checks in. He does not micromanage you; he offers strong, honest suggestions and lets you decide. He seems to intuitively know your limits and lets you struggle just enough to encourage you to ask for help, and when you do, he jumps in with the kind of reliability, grace, and efficiency reserved for concierges at fine hotels. That's not Mike's 'style', that's just Mike being Mike."

-Jon Bach, QE Director at eBay

How much does coaching cost?

Given the variety of the activities we will engage in, we don't often charge by the hour. Instead, we charge by the month. We work with you to figure out meeting schedules, onsite/offsite availability, and key goals for a given month/sprint. Once you become a client, you can call, instant message, or email your coach for additional coaching or support at any time. If we aren't immediately available, we can usually follow up within 24 hours.

Pricing will vary based on the size of the team and the scope of the engagement, but it's not uncommon for teams to pay somewhere between \$1,500 and \$4,000 a month. You don't pay in advance; you pay at the end of each month. If you're not happy with the coaching you've received, then you don't pay. Most engagements last around three months to get things started, but you have ultimate control of when we start and stop.

Contact us today to discuss how we can help you get a fresh perspective on ways to make your team more effective.