A Conversation with Howard M. Wiener, MSIA, CERM, PMP About Agile 2

Description

Interviewed by Greg Hutchins, PE, CERM

Iterative software development approaches have been around since the 1950s but many mark the beginning of what is commonly thought of as Agile as the creation of the Agile Manifesto, a document written by a number of software engineers in 2001, which contains 12 principles that serve as guidelines to how software projects should be conducted. Since then, professional services and product organizations have taken Agile and bent and twisted it into proprietary commercial forms for their own gain. Proponents evangelize various forms of Agile as superior and disparage others as lacking in rigor or efficacy. In fact, most of them are so close in structure and practice that if you were to strip away proprietary vocabulary and branding, you could easily mistake one for another. Thus, *Agile* turned into the *Agile Industrial Complex* and took on a life of its own. Orthodoxy and adherence to proprietary ceremonies and arbitrary standards of execution, of course supported by consulting, classes, exams and certifications, has become rampant.

What was the original Agile?

What the Manifesto was intended to address were a variety of problems with traditional approaches that often resulted in substandard or suboptimal results, if not outright project failures. The Waterfall approach consists of a sequence of phasesâ€”Requirements Definition, Design, Construction, Testing, Acceptance and migration to Productionâ€”that required completion of each phase before the next could be initiated. Common problems were:

- User requirements evolved or expanded over the course of the project. This is only to be expected; few users, even very experienced ones, can define ex ante what will best fill their needs. Seeing a system as it develops almost always brings new ideas into play. As software projects were lengthy affairs, the business environment could change, obviating some of the original requirements and creating needs for new ones.
- Development tools were pretty rigid. Change was expensive, time-consuming and problematic.
- Testing tools and practices were not as robust as todayâ€™s. Testing required substantial time and resources which were rationed like everything else.
- Projects were usually funded on an all-or-nothing basis, ordinarily during the companyâ€™s annual budget cycle. There was an expectation that projects would (a) deliver the specified functionality, (b) complete on schedule and (c) remain within budget. Changes to specifications that could impact these goals were, to say the least, frowned upon and often had negative career implications for those responsible for them. This has led to completion of numerous valueless projects that should have been scrappedâ€”the operation was a success, but the patient died.

Agile methods were intended to treat these problems by, among other things:

- Mandating autonomous self-organizing teams consisting of users and developers working collaboratively and emphasizing the teamâ€™s identity while downplaying individual contributorsâ€™.
- Minimizing management imposed from outside of teams.
- Eliminating much of the documentation produced in traditional approaches and replacing it with face-to-face collaboration.
- Decomposing systems into components that delivered working subsets of functionality and could be delivered in short bursts, say every two to four weeks.
- Providing for iteration to allow the system to evolve as users were able to see and touch it.

After nearly 20 years, it has become apparent that the *Agile Industrial Complex* is not producing the benefits that the authors of the Manifesto intended. A large percentage of systems development projects (estimated at 70%) were failing or at least underdelivering in the early aughts, and a similar percentage are still failing today.

So, what is Agile 2?

Agile 2 was developed by a worldwide group of 16 eminent practitioners with the intention of identifying and suggesting fixes for many of the things that are plaguing organizations employing current Agile methods and not getting what they want and expect from them. It consists of ten Principles, each supported by a number of problem statements and insights. It differs from Agile, as practiced by the Agile Industrial Complex, in some significant ways:

- It is targeted to facilitate the Digital Transformation that all companies need to undertake. It is specifically intended to help companies remake themselves to be competitive today and into the future.
- It focuses on delivering business outcomes, not working software, as the defining measure of success.
- It refuses to prescribe HOW to apply the Principles. It eschews commercialization, sharing insights to guide users to select elements of the approach that fit the work, culture and circumstances in which it will be applied.
- Understanding and accommodating team and individual dynamics is crucial to success. Elements of Agileâ€™s team-oriented approach tend to disenfranchise and inhibit contributions from team members that are introverted or prefer to write rather than talk at meetings. Agile 2 advocates that leaders explicitly observe, recognize and accommodate each team memberâ€™s working style in order to optimize their ability to contribute.
- Minimizing events that break team membersâ€™ concentration and produce little value. The Agile daily standup meeting, though intended to be no more than 15 minutes costs participants a multiple of that time in lost productivity. Team leadership should analyze information flow throughout the team(s), implement mechanisms that allow members to access information in a way that minimizes disruption to their concentration and restrict meetings to only those that truly demand participation of multiple members.
- Teams cannot be completely self-organized. Todayâ€™s componentized software is developed by numerous specialized groups and coordination is critical.
- Many forms of Leadership are required for success. Agile disdained leadership in favor of holacracy. Ultimately, it hasnâ€™t worked.

So, why does Agile 2 matter?

Agile 2 is intended to complement the processes and practices required to operate in a DevOps, fast-twitch product- and customer value-driven environment. Few, if any, organizations have achieved a mature, stable instance of these in a fully-realized form today, so the competitive challenge is exacerbated by the fact that companies will have to change the tires while driving the car on the highway. The fact is, however, that companies will have to learn how to work in a constant state of evolution and the Principles contained in Agile 2 are formulated and intended to prepare them to function effectively in this state.

Date Created
2021/02/16
Author
howardmwiener