Waterfall-Teamwork vs Agile-Teamwork – part 2

By Mike Richardson, Team Agility Practice Leader

In part 1 of this article last month (link) we looked at how the future of agile-teamwork is already here, in the field of agile-software-development. Yet many leaders, teams and organizations are still stuck in waterfall-teamwork. Waterfall is a term often used in traditional software development methodology and refers to the trickle down, stepwise and linear approach to project management. Over the past ten years, the notion of AGILE project management has been transforming the way stuff gets done better and faster in a non-linear and iterative methodology. "Waterfall" vs "Agile" contrasts two fundamentally different modes of teamwork and part 2 of this article will help you understand the difference.

Here's my summary for some of the major contrasts between Waterfall and Agile mindsets:

Waterfall Mindset	Agile Mindset
Gantt Chart (24 Month/Industrial Strength Project Plan)	Sprints & Scrums
Launch late and iterate never	Launch early & iterate often
Task is fixed, time is variable	Time is fixed, task is variable
Right first time	Wrong first time
Failure avoidance	Fail fast/small/early
Stages & Gates/Post-Mortem (critical-path-analysis/pacing/imposing impediments)	Removing Impediments/Regular Retrospectives (self determining teams/increasing velocity)
Knowing & Controlling (an exercise in futility/fallacy/fiction)	Not-Knowing & Not-Controlling (an exercise in learning & growing)
Learning by Thinking	Learning by Doing
Paperwork/Paperwork/Paperwork	Product/Demos/Real-Feedback

Agile mindsets are set free by a willingness to get comfortable being uncomfortable, with the discomfort of being wrong, of failure and of not-knowing. As a result, it becomes an increasingly high velocity process of learning by doing, Most crucially, it's highly iterative with daily scrum meetings (for the team to take stock and adjust every day) and short sprint-cycles (often only 1 or 2 or 4 or 6 weeks) at the end of which the team demo's real product to real people for real feedback.

Waterfall mindsets are held hostage by a need for knowing and control, for the comfort of failure avoidance. But it's an exercise in futility, laced with fiction (we make up a critical path based upon assumptions that are un-knowable and un-controllable) and fallacy (we project rates of progress that we think we can achieve before we have done anything). Most crucially, we create an industrial strength project plan and Gantt Chart perhaps out over a 24 month timeframe, which is front-end loaded with paperwork, paperwork and more paperwork!

One of my favorite and most illuminating contrasts between Waterfall and Agile is that between the "task is fixed, time is variable" mindset of Waterfall and "time is fixed, task is variable" mindset of Agile. I was once running an enterprise scale software company and we had just lost a strategic bid because the customer told us that our graphic-user-interface (GUI) was perceived as out-of-date. We were in trouble because I had just acquired the company into my division and this was a major setback. I got my team in a room and said, "You have 3 days to be back in this room telling us where you think you can be 3 weeks from now with a new demo GUI. In 3 weeks you will be back in this room showing us the demo and telling us where you can think you can be in 3 months with the real thing". I told them where the hurdles were (time is fixed) but not how high to jump (task is variable), which they self-determined. 3 months later they wowed us with the new GUI they had released and we were back in business. Imagine what answers I would probably have got if I had said, "We need a wow new GUI, how long will that take?" I didn't know it at the time but I had intuitively used an agile mindset not a waterfall mindset.

In his book, "Scrum: The Art of Getting Twice as Much Done in Half the Time", Jeff Sutherland puts the contrast this way:

"Traditionally, management wants two things on any project: control and predictability. This leads to vast numbers of documents and graphs and charts. Months of effort go into planning every detail, so there will be no mistakes, no cost overruns, and things will be delivered on schedule. The problem is that the rosy scenario never actually unfolds. All that effort poured into planning, trying to restrict change, trying to know the unknowable is wasted. Every project involves discovery of problems and bursts of inspiration. Trying to restrict a human endeavor of any scope to color-coded charts and graphs is foolish and doomed to failure. It's not how people work, and it's not how projects progress. It's not how ideas reach fruition or how great things are made. Instead, it leads to frustrated people not getting what they want. Projects are delayed, come in over budget, and, in too many cases, end in abject failure. This is especially true for teams involved in the creative work of crafting something new. Most of the time, management won't learn of the glide path toward failure until millions of dollars and thousands of hours have been invested for naught."

Sound familiar? By contrast, Jeff Sutherland goes on to explain Scrum, which is at the heart of Agile:

"Scrum embraces uncertainty and creativity. It places a structure around the learning process, enabling teams to assess both what they've created and, just as important, how they created it. The Scrum framework harnesses how teams actually work and gives them the tools to self-organize and rapidly improve both speed and quality of work. At its root, Scrum is based on a simple idea: whenever you start a project, why not regularly check in, see if what you're doing is heading in the right direction, and if it's actually what people want? And question whether there are any ways to improve how you're doing what you're doing, any ways of doing it better and faster, and what might be keeping you from doing that. Companies that still cling to tried-but-not-true ideas of command and control and that attempt to impose rigid predictability are simply doomed to fail if their competitors use Scrum. The difference is too great."

Take two head to head competitors, in any field of business. One takes a waterfall approach to strategy and execution and the other takes an agile approach to strategy and execution. Which one will you bet on will come out ahead? I would bet on the agile one for sure.

We still encounter the waterfall-thinking way too much in business. Here are some examples:

- **Waterfall Strategic Planning** (vs Agile Strategy Process) – a 3 year, 5 year or 10 year strategic plan? Really?
- **Waterfall Budgeting** (vs Agile Budgeting) – an annual Fiscal Year budget, tracking year-to-date this Fiscal Year vs year-to-date last Fiscal Year? Really?
- **Waterfall Goal-Setting** (vs Agile Goal-Setting) – an annual goal-setting cascade and annual appraisal? Really?
- And More:
    - **Waterfall Change Management** (vs Agile Change Management)
    - **Waterfall Leadership** (vs Agile Leadership)
    - **Waterfall Thinking** (vs Agile Thinking)
    - And so one

In an increasingly VUCA world**, waterfall anything doesn't work!** We help teams understand the mindsets, thinking and leadership of agile everything, to develop an enterprise agility advantage – read more next month in part 3 of this article.