When you ask me: what's agile, I'd like to say …

Paul Magnuson
October 2016

It happened again. Someone asked me about agile and even wanted to hang out long enough to understand what it is and why it would be good for education. I answered her, but didn't make the most of the opportunity. Maybe taking the time to write about an agile mindset in education is a good idea, if just to sort out my thoughts. And maybe next time I have the chance to bend someone's ear, I can be a bit more convincing.

Agile. It started when some software developers got together and put their ideas into as simple a document as they could. They called it a manifesto, which is a maybe a turnoff. It makes it sound as if these folks thought far too highly of themselves. Who writes manifestos these days? Then again, they did seem to boil down an approach to work into just a few general principles that turn out to apply far beyond their own area. So I'm giving them the benefit of the doubt on the whole manifesto thing and seeing how we can apply it to education.

I'll paste their four guiding points in below. But before I do that, indulge me for a second as I paint a few pictures of a non-agile mindset.

Can you picture a Dilbert cartoon where the boss asks for feedback and some poor newbie gives some? Only to be given the worst cubicle, schedule, and projects? If that strikes a chord with you, perhaps you have a feeling for a non-agile organization.

Or have you worked in an organization where someone higher up the org chart is allowed to state that "this is how things are" and the employees lower in the chart know that the best path to changing that mandate is probably to wait for that person to find a different job? That organization probably wasn't too agile.

Or have you been in a meeting where colleagues bring up questions which are all directed at the chair of the meeting, who is also the boss? The employees may have internalized that decision making rests solely with the boss, so why be asking others? If the chair, who is the boss, feels that he or she needs to provide the answer to all the questions, because knowing the answer is what bosses do, the organization is probably not terribly agile. Potentially terrible, certainly, but probably not agile.

In a non-agile world, there are ways to do things and ways not to do things. Status in the organization plays an important role in who makes decisions, so those in status positions disproportionately create the culture's rules of how things are done. Because the hierarchy is triangular, with the greatest status on top, significant input is accepted from a small minority of people. Unfortunately it is the entirety of everyone in the organization that needs to understand the whats, whys and hows to get done what needs doing.

An agile mindset frees us, or might free us a bit - it's worth a try! - from the scenarios above. Here's what the writers of the manifesto agreed on. Remember, they were thinking about software development. You'll have to do some translating into your own setting if you aren't C++ing or Rubying.

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

- From the Agile Manifesto (2001)

Just quit reading for a moment and think about these four points. There are only four, give it a minute or two.

…

OK. Here's my summary.

People first. People who make stuff that works. People who collaborate with the people who will use what they're making. People who listen to and respond to feedback along the way.
We could play with the opposite again to highlight the differences of an agile and non-agile mindset.

Processes first? Processes that are really well documented. Negotiation of contracts in order to support one really big plan.

Now, neither the authors of the manifesto nor I are saying that processes, documentation, negotiation and planning are unnecessary. Rather, the manifesto is saying that people, working products, collaboration and ongoing feedback is *more* necessary - that these factors should come first.

Let's see how these principles apply in school.

- Individuals and interactions over processes and tools
    - Interaction of Administrators and teachers, teachers and teachers, teachers and students, and students and students working face to face, are valued far more than the types of tools that are used (consider for example the common admonition to avoid tech for tech's sake)
    - Administration supporting the people to people soft skills of teachers, not more pd to understand the latest software, the latest textbook series, or the latest tests and accompanying test prep.
- Working software over comprehensive documentation
    - Student learning over detailed plans about what students are going to learn - what would that mean for curriculum?
    - Student learning over time spent documenting grades and behavior.
- Customer collaboration over contract negotiation

    - An emphasis on listening to students and incorporating their ideas and interests rather than adhering to the syllabus, to the written curriculum...

- Responding to change over following a plan

    - Starting with the assumption that a written curriculum (and a class syllabus) will begin to be inaccurate from the minute they are first published.

    - Believing so strongly in feedback that formative feedback from students - on their learning and on the teacher's teaching - are routinely collected and discussed, with students.

So you might not agree with each of these and you might need to catch me in a free moment to talk more about them. I hope you do. Putting it down in writing might have made me a little more prepared for that conversation than I was last time.

Paul can be contacted at pmagnuson@las.ch.