Questions and Answers from audience

Question: I am just starting to get involved in the PM world. I have run many projects in the past, however never utilized any of the methodologies. How do you recommend I get my feet wet, whilst being still employed?

Answer: I have a few recommendations:

- Read Research, Books/Attend Webinars - There are a number of books, blogs, webinars, etc. that you can use to build your project management knowledge.
- Attend classes - The University provides a number of courses in project management that provide you with a solid understanding of the fundamentals of project management.
- Volunteer as Project Manager for a Non-Profit or Community Organization - Non-Profits or Community Organizations are always looking for volunteers and usually have a number of projects that they need completed. This is a great way to build some experience in a non-threatening environment.
- Shadow an Experienced Project Manager - Find an experience project manager in your organization and ask to shadow them on a project, either as pure career development experience or by serving as the project administrator. This will allow you to see a project manager in action and compare their approach and experience to those presented in the literature or at a class.

Question: does the 4 week sprint time include the testing phase or is this in addition to the 4 week sprint?

Answer: the sprint should include all of the activities necessary to get the software to "Done" including testing. Some teams include a "hardening" sprint prior to release to complete a final set of regression and acceptance testing, but care needs to be taken to not make this sprint a "dumping ground" for additional features or an excuse for not testing incrementally as the project progresses through each sprint.
**Question:** What is the difference between a scrum master and a service delivery manager?

**Answer:** The answer depends on a large on how you define the responsibilities of a service delivery manager; however, a Scrum Master is responsible for the following:

- Serving the Team - No one reports into the Scrum Master
- Removing Barriers
- Facilitating the Team through the Scrum / Development Process

In my experience a service delivery manager has responsibilities beyond those listed above. However, once again, it depends on how you define the role.

**Question:** I apologize for missing the first portion of the session... I missed the role of data modelers (LDM & PDM/DBA) and the impact to the length of a sprint. In particular with respect to an Enterprise Data Warehouse (foundation & performance layers).

**Answer:** You raise a good question regarding the role of a data modeler within an Agile Team. While we did not directly address this role during the webinar, data modelers fundamentally share the same responsibility that all members of the team have - to get to "done" and deliver working software that people want to use. In practice, data modelers often assist the product owner and the team by capturing data rules, definitions, etc. that get added to the product backlog as "constraints". They use these constraints to help build light-weight models for the Team to aid in the development of software in an attempt to meet the aforementioned goal.

**Answer:** In addition to what Mike answered below, keep in mind any Agile approach is different. For that reason there is no one for one correlation with traditional roles in any of the Agile approaches.

The goal is for the right people, with the right skills to approach the work together as a team to achieve business results through working product as quickly as they can. Agile asks that the team be empowered to come up with how they achieve this and does not prescribe who does what and when like a Waterfall methodology does, for example.
**Question:** We just moved to a matrix organization with Service Owners and Line Managers. Will agile work with this type of matrix?

**Answer:** This is one of the most frequently asked questions we receive in teaching Agile methods. The premise of Agile teams is focus, dedicated, committed teams. You probably have already guessed then that the answer is not really; Agile methods do not typically work well in a matrixed environment as it's a conflicting premise to any Agile approach.

Agile methods are able to produce higher quality product faster than traditional methods (which typically involve matrixed teams) as team members are allowed to focus on one thing before moving onto something else.

If in your matrix scenario, it is only for reporting relationships and people will be devoted to an initiative 100%, there is still some risk that at the end of the "project" they will be moved onto another application or project. Agile methods focus on Products vs. Projects. This allows the team's productivity to go up over time as they stay dedicated to a particular product with its enhancements, production fixes, etc.

**Question:** How do you deal with a lot of changing requirements with managers who don't understand how and why there are so many changes (overly waterfall and old school)? It’s very high stress.

**Answer:** Agile methods embrace change for the customer’s competitive advantage. With each new Sprint, the customer has the opportunity to introduce something new. This is not a license to "pile on" to the team's plate. It's more about "plate management". Something comes onto the plate, something has to come off. So it's not a license for change to occur whenever anyone wants it to - it's introduced at the beginning of an iteration or Sprint.

There is only a particular amount of work that can fit into each Sprint. This disciplined approach to change with each iteration is actually about creating sustainable pace, as opposed to piling onto the plate without taking anything off and still expecting the same delivery date. The result is a more consistent delivery mechanism that ensures we're delivering what the customer asked for based on their involvement in the process.
**Question:** How do PMs & BAs decide when to use Waterfall vs. Agile Project Methodology? Isn't this just the pendulum swinging back to the way software development was in the US in the 80's and 90's?

**Answer:** It certainly seems that way doesn't it? A Waterfall approach is risky since we don't involve the users until the end when we may be out of time and money to respond to anything that is not correct or to anything that has changed in the customers marketplace, with their regulatory compliance needs, etc.

Agile methods tend to pay off when the problem that is trying to be solved is deemed Complicated or Complex. If the problem is Simple, any method or approach may work just find for delivering what was requested. If we don't have a lot of agreement or certainty, an Empirical process, such as Agile, that allows us to "learn as we go" works best. If we have high certainty and solid agreement, a Defined approach may work just fine.

The questions to ask to determine if your problem is Simple, Complex or Complicated involves the number of stakeholders, the type of technology, the environment, the people doing the work and the deliverables. The more likely you are to have change, or have unknowns or new technology, the more complicated the problem becomes. A Defined process such as Waterfall will not provide the opportunity to "learn as you go" inspecting and adapting as necessary like Agile methods will.

**Question:** What happens when the end of the project is nearing, but the final sprint won't complete the product backlog?

**Answer:** With Agile methods, it is not assumed that we are going to complete every item on the Product Backlog. The Product Backlog is not an artifact that is "signed off" and that does not change like Requirements in a Waterfall approach.

As items are added to the Product Backlog, other items move down the list or may move up the list or may even be removed from the list. It is a living, evolving, emerging artifact.

The team commits only to doing the highest priority items on that list that can fit into the iteration or Sprint. That means that the Product Owner or Customer must choose which items will bring the most value - the goal is not to deliver every single item. Agile manages Value instead of Scope. So the goal is only to deliver those items that will realize the most value.