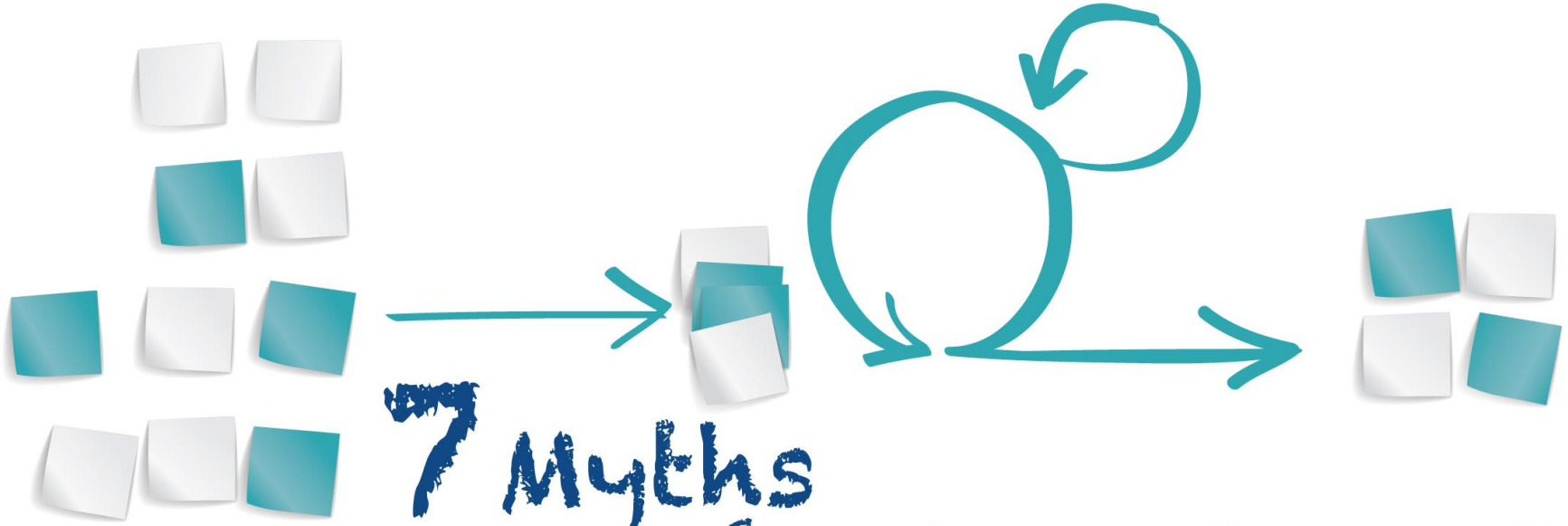
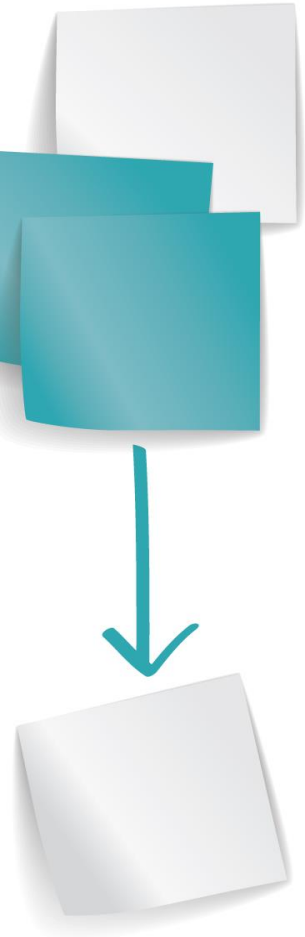
intelliware
software development

Agile



7 Myths of Agile Development



Introduction

There are dozens of myths about Agile development. But before jumping into specific misconceptions, let's have a look at some common business challenges:

For senior-level execs: do you value revenue growth or cost containment?

For project managers: do you value team efficiency or effectiveness?

For developers: do you value code quantity or quality?

In each scenario, you probably struggled to make a choice given that your two options were not mutually exclusive.

Posing the question this way creates a false dilemma since you likely value both options but to varying degrees. So the better question is, of the two options, which do you value *more*?

Introduction (continued)

The Agile Manifesto evolved through dilemmas like those just mentioned. Often two opposing approaches, such as responding to change versus following a plan, were deliberated upon until the authors of the Manifesto decided that it would be best if they valued one approach more than the other, instead of choosing one over the other.

Unfortunately, many of the myths about Agile are based upon straw man fallacies: oversimplifications of arguments to make them easier to attack. For example, the Agile Manifesto states that Agile values working software over comprehensive documentation. Agile detractors often oversimplify this idea as an either/or state: working software or comprehensive documentation.

What follows are the most common Agile Myths and a rebuttal to these false claims.

Figure 1: The Agile Manifesto
Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, and Dave Thomas.

Myth #1

Agile development is a methodology

While not as common as the other 6 myths, this one needs to be addressed first. Agile development is not a methodology – it is a set of shared values and principles that guide a set of technically rigorous development methodologies. These methodologies include, but are not limited to: Scrum, Kanban, XP (Extreme Programming) and Lean Software Development.

Authors of “The Art of Agile Development”, James Shore and Shane Warden, argue, “Agile development is a philosophy. It’s a way of thinking about software development.” We view Agile as a philosophy that addresses the forces at work in organizations that are striving to deliver high quality software applications to the customer and their often evolving needs.

© 2013 Intelliware Development Inc.

Myth #2

Agile is undisciplined

This myth is based on a perceived lack of process. Agile places greater value on individuals and interactions than on processes and tools. Therefore, some conclude illogically that Agile lacks process and therefore discipline.

This myth comes in many flavours: “Developers get to do what they like.”[2] Agile = anarchy.[3] Agile is “cowboy programming”.[4] Agile means “code and fix”.[5]

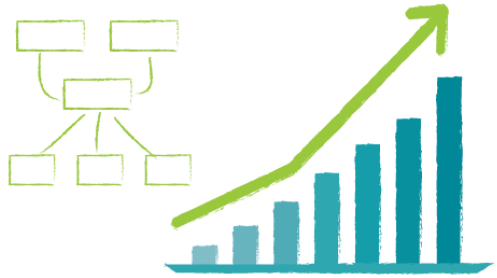
Incomplete Agile transitions help keep this myth alive, as some companies have adopted the easier parts of Agile while ignoring the harder parts.[6] This is sometimes referred to as “Agilefall”.

There is process to Agile, though it isn’t the one-track, sequential process of traditional development. Agile processes often occur in parallel with one another and are repeated.[7] In fact, many argue that Agile actually requires a greater level of discipline than more traditional approaches.[8,9] In our experience the ‘agility’ in application delivery is enabled through a very disciplined approach to technical application development practices.

© 2013 Intelliware Development Inc.

Myth #3

Agile has no planning



"Plans are worthless, but planning is everything."[10] *- Dwight D. Eisenhower*

Eisenhower made the above comment to a National Defense Committee in the context of planning for emergencies. But there is a reason why it has been quoted by Agile proponents: both national defense emergencies and end-user software needs are unforeseen.

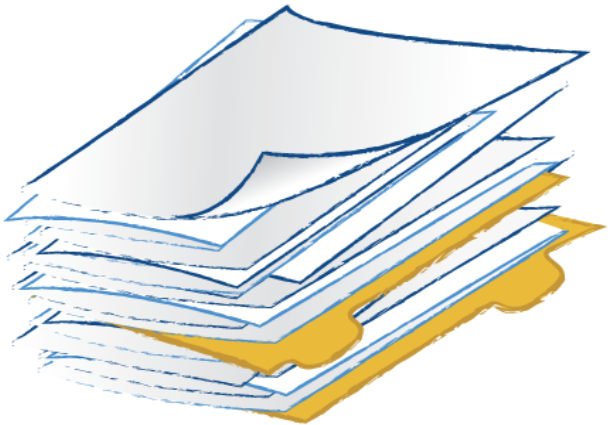
To be agile is to be able to move quickly and easily. Planning can inhibit agility. Some illogically conclude that all planning inhibits agility. As English author Lewis Carroll originally said, "if you don't know where you are going [no plan], any road will get you there." The right amount of planning is essential to properly guide your 'agility'.

In Agile development, Big Design Up Front (BDUF) is avoided; planning occurs throughout the development cycle and is spread across the entire team.[11] It avoids the situation where overly detailed plans made at the start of a project become out of sync with the technical and business needs as the project progresses.[12] Agile aims to work the plan, not work to the plan. The result is a constant focus on business value.

Throughout development, the team adapts to make sure the plan reflects the current needs of the customer. Therefore, Agile welcomes the changes that are inevitable in software development and plans accordingly. Release and iteration plans detail both what needs to be done and how it will be done. Delivering working software throughout the development process at agreed upon deadlines requires planning.

© 2013 Intelliware Development Inc.

Myth #4
Agile has no documentation



The Agile founders met to create an alternative to document driven development.[13] They did not set out to remove documentation from software development. Agile simply places more value on working software than on *comprehensive* documentation because of the dynamic nature of software development. As requirements are modified, the development changes course and the software evolves. This does not preclude any development team from generating as much documentation as the project requires. Indeed, the natural process of Agile development tends to generate a greater amount of (and more accurate) documentation than BDUF methodologies.

Comprehensively documenting a system (particularly when it's done "up front") can be a poor use of time since changing requirements renders documents obsolete or inaccurate. Also, there is a risk of misunderstanding between the customer and developer when relying on written documentation to express software requirements. However, there are situations (documenting interfaces between systems, for instance) in which documentation is absolutely required. There is nothing inherent in Agile that prevents you from creating as much documentation as your project requires, especially if the customer values it. Agile just suggests you be smart about it and that documentation not take on a life of its own. Documentation that provides value to the customer is much different than documentation that is produced for the sake of documentation or to support a process.

To quote Mike Cohn, in a Waterfall approach, "Customers will get the developers' interpretation of what was *written down,* which may not be what they *wanted.*"[14] The iterative delivery of working software effectively replaces much, though not all, of the comprehensive upfront requirements documentation. A picture is worth a thousand words. Working software is worth even more.

© 2013 Intelliware Development Inc.

Myth #5

Agile has no upfront design/architecture

Agile stresses a simplification of upfront design, not the elimination of upfront design. As argued by Robert C. Martin, one of the founders of the Agile Manifesto, Big Design Up Front (BDUF) is “harmful” but little upfront design (LUFD) is “absolutely essential”.[15]

The harm from BDUF can take shape in an overly complex product, which is a typical outcome for projects developed using the Waterfall approach. Agile development stresses simple upfront design to focus on the foundation and general structure of the software. Agile developers avoid building software features that may or may not be needed; they build for the current need and get feedback in the iterative delivery of software to the client.

In XP, an Agile method, there is a principle called YAGNI (You Aren’t Gonna Need It) to help focus on designing at the right time in the process.[16] As with drawing, Agile recommends beginning with a sketch to explore the presentation of a concept. If the concept is validated, then the details are added.

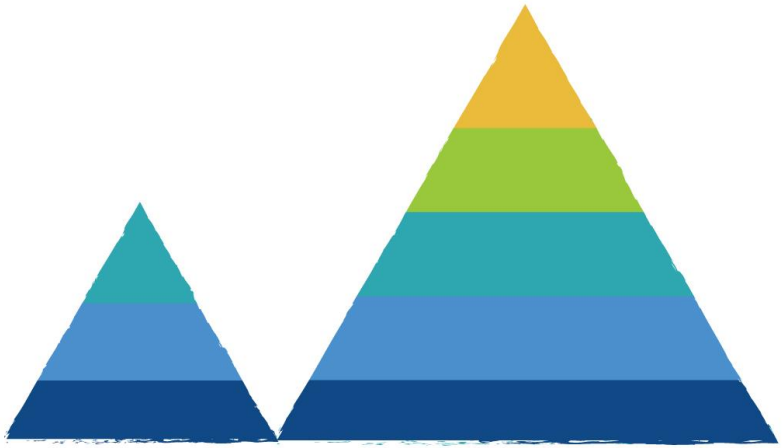
© 2013 Intelliware Development Inc.

Myth #6

Agile does not scale

Scaling software development is difficult, regardless of approach. Some believe that Agile may work fine for small projects but not for large, complex projects.

Agile encourages breaking large, complex projects into many small, manageable pieces. This means that it can indeed scale – even for big projects. Of course, it really is a matter of approach. Some Agile practices need to be tweaked for the realities of large projects. For example, the larger the team, the shorter the development cycle should be. By keeping the development cycles short, the project remains a series of small, manageable projects. As well, face-to-face conversations will likely be limited in large projects, so technologies that enable the closest representation of this form of communication, such as videoconferencing, should be used. Finally, since continuous integration can be a significant challenge on large Agile projects, there should be an integration team.[17]



© 2013 Intelliware Development Inc.

Myth #7

Agile is just another fad

A fad is, by definition, short-lived. Agile has been around for over a decade. Some of the central tenets have been in practice since the ‘70s. Agile has been around for too long to be a fad, especially when one compares it to the relatively short history of software development. This myth may be propagated by those who comprise the laggards of Agile adoption – many of whom tend to resist change.

Agile, as a philosophy (not a methodology), was created in response to the inherent complexities of software development; therefore, as long as there is software development, Agile will exist.

An alternate explanation may be that Agile is still in a “hype cycle” and therefore, still subject to possibility of being a fad. In Gartner’s 2010 report, “Hype Cycle for Application Development”, Agile Development Methods were considered as, “sliding into the trough (of disillusionment)”.[18] That may sound ominous, but for those who believe in this model, it is merely a growing pain. Java went through this trough before reaching its plateau.[19] Agile continues to develop. In 2012, Gartner also stated that Agile’s move through the Trough of Disillusionment is a “normal part of any IT trend that is going mainstream” and that “the long term trend of agile is working well in more and more companies, so the future of agile is still promising.”[20]

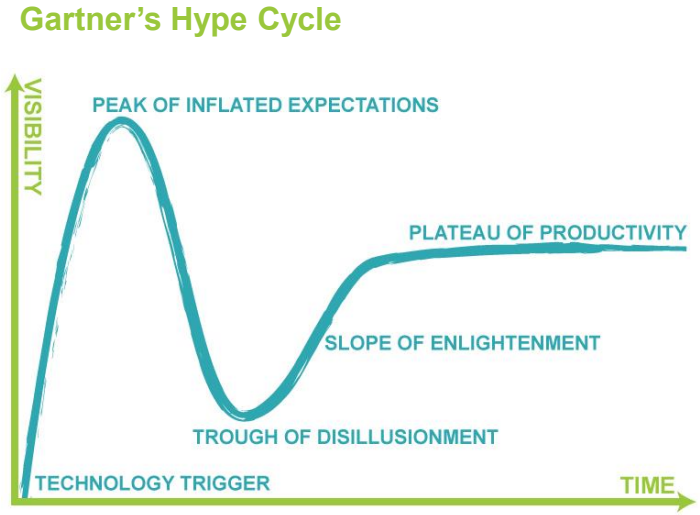


Figure 4: Gartner's Hype Cycle

© 2013 Intelliware Development Inc.

Conclusion

Myths and criticisms about Agile software development abound. Don't let them impede your progress. Agile is not a fad – the forces that have led us to Agile are not going away. Regardless of your proficiency with Agile, it is important to be mindful of these common myths and criticisms because Agile requires organizational buy-in. Therefore, you will need your colleagues to understand Agile, which will likely require you to debunk a few of these myths yourself. If you can succeed at this task, you will increase your chances of fully realizing the business benefits of Agile, such as faster time-to-market, higher quality software, lower costs and a greater ability to adapt to changing priorities.

© 2013 Intelliware Development Inc.

About Intelliware Development Inc.

Intelliware is a custom software, mobile solutions and product development company headquartered in Toronto, Canada. Intelliware is a leader in Agile software development practices which ensure the delivery of timely high quality solutions for clients. Intelliware is engaged as a technical partner by a wide range of national and global organizations in sectors that span Financial Services, Healthcare, ICT, Retail, Manufacturing and Government.

 /company/intelliware-development-inc-

 /intelliware.inc

 /intelliware_inc

 /bit.ly/GooglePlusIntelliware

www.intelliware.com

© 2013 Intelliware Development Inc.

Sources

[1] Shore, James, Warden, Shane. *The Art of Agile Development*. Sebastopol: O'Reilly Media, Inc., 2008. Print.
[2] Kelly, Allan. "Top Twelve Myths of Agile Development." *The Agile Connection*, TechWell Corp. 27 Mar, 2013. Web.
[3] Löffler, Marc. "7 Agile Myths". *The Agile Zone.* 29 Jan, 2013. Web.
[4] "O'hEocha, Colm. "Agile – Adoption: Agile Myths". *AgileInnovation Ltd.* www.agileinnovation.eu. 2010. Web.
[5] Holler, Robert. "Five Myths of Agile Development". *VersionOne*. 2010. Web.
[6] Rasmusson, Jonathan. "Agile Myths". *Agile in a Nutshell.* Web.
[7] Gregory S. Smith, "What an Agile Process Looks Like". CIO.com. 23 Jan., 2008. Web.
[8] Scott M. Ambler and Matthew Holitza. *Agile for Dummies*. Hoboken: John Wiley & Sons, 2012. Print.
[9] Holler, Robert. "Five Myths of Agile Development". *VersionOne*. 2010. Web.
[10] From a speech to the National Defense Executive Reserve Conference in Washington, D.C. (November 14, 1957) ; in *Public Papers of the Presidents of the United States, Dwight D. Eisenhower, 1957*, National Archives and Records Service, Government Printing Office, p. 818 : ISBN 0160588510, 9780160588518
[11] Kelly, Allan. "Top Twelve Myths of Agile Development." *The Agile Connection*, TechWell Corp. 27 Mar, 2013. Web
[12] Holler, Robert. "Five Myths of Agile Development". *VersionOne*. 2010. Web.
[13] Highsmith, Jim. "History: The Agile Manifesto". *Agilemanifesto.org.* 2001. Web.
[14] Mike Cohn. *User Stories Applied For Agile Software Development.* Boston: Peasron Education, 2004. Print.
[15] Martin, Robert C. ("Uncle Bob"), "The Scatology of Agile Architecture". *Uncle Bob Consulting LLC*. April 25, 2009. Web.
[16] Rasmusson, Jonathan. "Agile Myths". *Agile in a Nutshell.* Web.
[17] Jutta Eckstein and Nicolai Josuttis. *Scaling Agile Processes: Agile Software Development in the Large.* Agility Days 2002. Web.
[18] Janes, Andrea, and Succi, Giancarlo. "The Dark Side of Agile Software Development". *Darkagilemanifesto.org*. Free University of Bolzano/Bozen. 2012. Web.
[19] Janes, Andrea, and Succi, Giancarlo. "The Dark Side of Agile Software Development". *Darkagilemanifesto.org*. Free University of Bolzano/Bozen. 2012. Web.
[20] Wilson, Nathan. "The Trough of Disillusionment". *Gartner*. 27 July, 2012. Web.

© 2013 Intelliware Development Inc.