JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Integrating Security in Agile Software Development

Student Name: Sonali Arya

CSE at DPG Institute of Technology and Management

Teacher Name: Ms. Smriti Dwivedi

Abstract:

Agile is popular approach in software development. Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross- functional teams. Agile methodologies provide better results in return on investment, quality, team morale stakeholder satisfaction and delivery time. Security has cannot be given the attention it needs when developing software with Agile methods. Now a days there are strong regulatory and privacy requirements to protect private data, security must be treated as a high priority.

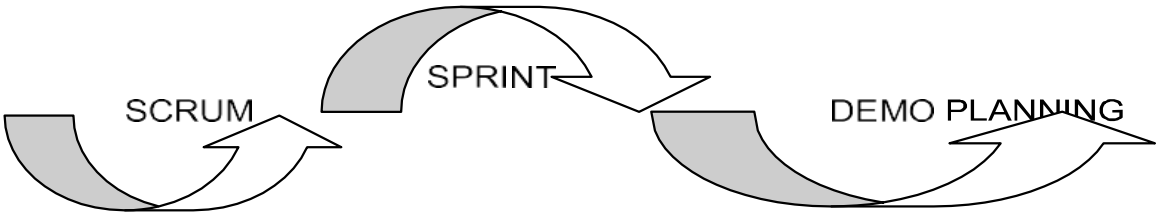
Agile software development is a set of techniques for building intended software with fewer errors and better predictability. Each technique or approach is intended to address well-known weaknesses in traditional 'Waterfall' development: complexity, poor communication, and infrequent code validation.

In this paper we will discuss on agile methodologies security, extreme programming, agile manifesto, future aspects and technological development of agile computing.

Keywords: Agile, Extreme Programming, Scrum, Agile Development, SDLC.

Introduction:

Microsoft use Agile software development and management methods to build their applications. Now a days highly inter connected world, where there are strong regulatory and privacy requirements to protect private data security must be treated as a high priority. The strength of Agile is that it can save organization significant amounts of development time and money, while still allowing them to deliver high quality software. Agile practices are related to improved product quality customer satisfaction, and developer productivity than traditional waterfall practices. Agile practices have a significant impact in developing software in recent few years. Agile cycle includes includes Scrum, Sprint and demo-planning. Agile cycle is short and there is no directed mapping from Agile cycles to Waterfall stages.



The Fundamental characteristics of software development approaches:

1. Specification, design and implementation are interleaved.
2. The system is developed in a series ofversions.
3. User interfaces are often developed in an interaction development environment that allows for quick redesign.

Agile methods are not well suited for critical and complex system.
Things that may come the way of and development:

1. Customers are not able or willing toparticipate
2. Team members are not comfortable for intense involvement
3. Prioritizing changes can be difficult
4. Maintaining simplicity requires extra work.

Agile Testing VS Waterfall Testing

Agile Testing Manifesto

Agile Testing Values

Principles Behind Agile Testing Manifesto

Agile Manifesto:

Agile development refers to any development process that is aligned with the concepts of the Agile Manifesto. The Manifesto was developed by a group fourteen leading figures in the software industry, and reflects their experience of what approaches do and do not work for software development. The Agile Manifesto was created as an alternative to document-driven, heavyweight software development processes such as the waterfall approach. The four core values of agile software development as stated by the Agile Manifesto emphasize.

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation.
3. Customer collaboration overcontract negotiation.
4. Responding to change over following a plan

The agile manifesto principles are-

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable Software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale
3. Business people and developers work together daily throughout the project.
4. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
5. The best architectures, requirements and designs emerge from self-organizing teams.Simplicity, the art of maximizing the amount of work not done.is essential.
6. Continuous attention to technical excellence and good design enhances agility.
7. Agile processes promote sustainable development. The sponsors, developers and users should be able to maintain a constant pace indefinitely.
8. Working software is the primary measure of progress.
9. The most efficient and effective method of conveying information with and within a development team is face-to-face conversation

The 12 principles laid down in the Agile Manifesto have been adapted for managing a variety of business and IT-related projects, including business intelligence].

They include:

1. Satisfying 'customers' through early and continuous delivery of valuable work.
2. Breaking big work down into smaller components that can be completed quickly.
3. Recognizing that the best work emerges from self-organizing teams.
4. Providing motivated individuals with the environment and support they need and trust them to get the job done.
5. Creating processes that promote sustainable

6. Maintaining a constant pace for completed work.
7. Welcoming changing requirements, even late in a

8. Assembling the project team and business owners on a daily basis throughout the project.
9. At regular intervals, having the team reflect upon how to become more effective, then tuning and adjusting behaviour accordingly.
10. Measuring progress by the amount of completed

11. Continually seeking excellence.
12. Harnessing change for competitive advantage.

Future Prospects

The IT industry is now opening its eyes to the age-old industrial practices borrowed from the Toyota production system (TPS), the 3M company, non-IT industries, like healthcare and finance, have started embracing agile concepts faster than others. Agile methods do not create secure code, and on further analysis, the perception is reality. There is very little "secure agile" expertise available in the market today.

1. This needs to change but the only way the perception and reality can change is by actively taking steps to integrate security requitements into agile developing methods.
2. Some of the future aspects can be seen as follows:
3. From developers' point of view the quality of the developers is important.
4. From customer point of view agile, methods emphasize continues customer involvement, while traditional methods expect periodic customer involvement at formal millstone.
5. From requirement point of view, agile methods suggest they work better than traditional methods when requirements are volatile, much greater than 1% per month.
6. From architecture point of view, good architectures facilitate the addition of future functionality because they have, through planning, anticipated that functionality to a useful degree.
7. From re factoring part of view refectory is revising and simplifying the design or implementation of a software product without disturbing its functionality
8. From size point of view, agile methods can be adopted to larger (50-250 people) projects.
9. From balancing and risks point of view total risk exposure in project where a combination of agile and plan-driven methods are appropriate.

This is adapted from agilemanifesto.org and it might look a little silly to copy everything from there and just replace the term "development" with "testing" but here it is for your refreshment.

Principles Behind Agile Testing Manifesto

Behind the Agile Testing Manifesto are the following principles which some agile practitioners, unfortunately, fail to understand or implement. We urge you to go through each principle and digest them thoroughly if you intend to embrace Agile Testing. On the right column, the original principles have been re-written specifically for software testers.

We follow these principles:	What it means for Software Testers:
1. Our highest priority is to satisfy the customer through the early and continuous delivery of valuable software.	1. Our highest priority is to satisfy the customer through the early and continuous delivery of high-quality software
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	2. Welcome changing requirements, even late in testing. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.	3. Deliver high-quality software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.	4. Business people, developers, and testers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done	5. Build test projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is a face-to-face conversation.	6. The most efficient and effective method of conveying information to and within a test team is a face-to-face conversation.
7. Working software is the primary measure of progress.	7. Working high-quality software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	8. Agile processes promote sustainable development and testing. The sponsors, developers, testers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.	

10. Simplicity–the art of maximizing the amount of work not done–is essential. 11. The best architectures, requirements, and designs emerge from self-organizing teams. 12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.	9. Continuous attention to technical excellence and good test design enhances agility. 10. Simplicity–the art of maximizing the amount of work not done–is essential. 11. The best architectures, requirements, and designs emerge from self-organizing teams 12. At regular intervals, the test team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

Future Prospects

The IT industry is now opening its eyes to the age-old industrial practices borrowed from the Toyota production system (TPS), the 3M company, non-IT industries, like healthcare and finance, have started embracing agile concepts faster than others. Agile methods do not create secure code, and on further analysis, the perception is reality . There is very little "secure agile" expertise available in the market today.

1. This needs to change But the only way the perception and reality can change is by actively taking steps to integrate security requitements into agile developing methods.
2. Some of the future aspects can be seen as follows:
3. From developers point of view the quality of the developers is important.
4. From customer point of view agile, methods emphasize continues customer involvement, while traditional methods expect periodic customer involvement at formal millstone.
5. From requirement point of view, agile methods suggest they work better than traditional methods when requirements are volatile, much greater than 1% per month.
6. From architecture point of view, good architectures facilitate the addition of future functionality because they have, through planning, anticipated that functionality to a useful degree.
7. From re factoring part of view refectory is revising and simplifying the design or implementation of a sofMare product without disturbing its functionality
8. From size point of view, agile methods can be adopted to larger (50-250 people) projects.
9. From balancing and risks point of view total risk exposure in project where a combination of agile and plan-driven methods are appropriate.

Conclusion

Secure software comes from interdependently verifying field's integrated application created and by secure or insecure development as completely accurately and affordably as possible. Agile is all about being more responsive and better meeting customer requirements and these days nobody can credibly a gue that security is not key requirement. Agile approach is one of the most popular software development approach and methodologies using this appropriate need to be modified to adapt the increase threats and vulnerabilities information system.

Extreme programming is very helpful when you have enough resource and budget to invest on big project. It will improve code quality and help you to protract in the market. Agile has been criticized for lacking security due to consideration of security throughout the agile development life cycle and absence of the dedicated response person, having a fair knowledge of software security with defined responsibilities.

References:

1. Shahram Jalaliniya, Farzaneh Fakhredin, "Enterprise Architecture and Security Architecture Development", Master Thesis, Department of Informatics, Lund University, June 2011, PP.1 - 95.
2. Heiko Tillwick, Martin S Olivier, "A Layered Security Architecture: Design Issues", in Proceedings of the Fourth Annual Information
3. Ken Schwaber, Mike Beedle, "Agile Sohware Development with Scrum", Prentice Hall, 2001, pp. 89-94.
4. Sutherland, Jeff. "Scrum Web Home Page: A Guide to the SCRUM Development Process". Jeff Sutherland's Object Technology Web Page, 1996 //www.tiac.net/users/jsuth/scrum/index.html>
5. Schwaber, Ken. "Controlled Chaos: Living on the Edge."American Programmer, April 1996.
6. Aberdeen Group. "Upgrading To ISV Methodology For Enterprise Application Development. Product" Viewpoint 8:17, December 7, 1995.
7. Gartner, Lisa. "The Rookie Primer". Radcliffe Rugby Football Club, 1996 //vail.aI.arizona.edu /rugby/rad/rookie_primer.html>
8. Pittman, Matthew. "Lessons Learned in Managing Object-Oriented Development".IEEE Software, January, 1993, pp. 43-53.
9. Booch, Grady. "Object Solutions: Managing the Object-Oriented Project". Addison-Wesley, 1995. 11 Graham, Ian. Migrating to Object Technology. Addison-Wesley, 1994.
10. www.findwhitepapers.com
11. www.cs.umd.edu
12. www.techbeacon.com
13. www.microsoft.com/sdI
14. blog.venturepact.com
15. www.sans.org
16. www.techweII.com
17. www.design-reuse.com
18. Dave Shackllford, "Integrating Security into Development, No Pain Required", September 2011, ASANS Whitepaper.
19. David A. power, "Software development: Effective practices and Federal challenges in Applying agile Methods". July 2012, GAO -12- 681.