Global Journals LaTeX JournalKaleidoscope™
Artificial Intelligence formulated this projection for compatibility purposes from the original article published at Global Journals. However, this technology is currently in beta. *Therefore, kindly ignore odd layouts, missed formulae, text, tables, or figures.*

Comparative Study on Agile Software Development Methodologies

A B M Moniruzzaman[1] and Dr. Syed Akhter Hossain[2]

[1] Daffodil International University

Received: 6 December 2012 Accepted: 3 January 2013 Published: 15 January 2013

Abstract

Today?s business environment is very much dynamic, and organizations are constantly changing their software requirements to adjust with new environment. They also demand for fast delivery of software products as well as for accepting changing requirements. In this aspect, traditional plan-driven developments fail to meet up these requirements. Though traditional software development methodologies, such as life cycle-based structured and object oriented approaches, continue to dominate the systems development few decades and much research has done in traditional methodologies, Agile software development brings its own set of novel challenges that must be addressed to satisfy the customer through early and continuous delivery of the valuable software. It?s a set of software development methods based on iterative and incremental development process, where requirements and development evolve through collaboration between self-organizing, cross-functional teams that allows rapid delivery of high quality software to meet customer needs and also accommodate changes in the requirements. In this paper, we significantly indentify and describe the major factors, that Agile development approach improves software development process to meet the rapid changing business environments. We also provide a brief comparison of agile development methodologies with traditional systems development methodologies, and discuss current state of adopting agile methodologies.

Index terms— agile, traditional methods, agile adoption, SCRUM, XP.

Introduction lot of people have been asking the question "What is Agile Software Development?" and invariably they get a different definition depending on who they ask. Here's a definition that conforms to the values and principles of the Agile Manifesto **??**1]. An iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner by self-organizing teams within an effective governance framework with "just enough" ceremony that produces high quality solutions in a cost effective and timely manner which meets the changing needs of its stakeholders **??**6].Agile software development is actually a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between selforganizing, cross-functional teams **??**4]. In 2001, the "agile manifesto" was written by the practitioners reveals which items are considered valuable by ASDMs **??**1]. As shown in Table **??**.

Table **??** : Agile Manifesto (Source: [1]) a) Research Review Agile software development (ASD) is major paradigm, in field of software engineering which has been widely adopted by the industry, and much research, publications have conducted on agile development methodologies over the past decade. The traditional way to develop software methodologies follow the generic engineering paradigm of requirements, design, build, and maintain. These methodologies are also called waterfall-based taking from the classical software development paradigm. They are also known by many other names like plan-driven, (Boehm and Turner, 2004), [39]; documentation driven, heavyweight methodologies, and big design upfront, (Boehm, 2002), [16]. Boehm and

Phillip [72] report that during their project development experience, requirements often changed by 25% or more. Due to constant changes in the technology and business environments, it is a challenge for TSDMs to create a complete set of requirements up front [26]. **??**illiams and Cockburn,[18] also mentioned that one of problems of TSDMs is the inability to respond to change that often determines the success or failure of a software product. The agile approach to software development is based on the understanding that software requirements are dynamic, where they are driven by market forces **??**Fowler, Title 2002;); [16], [36]. Agile systems development methods emerged as a response to the inability of previous plan-driven approaches to handle rapidly changing environments (Highsmith 2002), [55]. Williams and Cockburn [18] state that agile development is "about feedback and change" , that agile methodologies are developed to "embrace, rather than reject, higher rates of change".

Agility is the ability to sense and response to business prospects in order to stay inventive and aggressive in an unstable and rapidly shifting business environment (Highsmith, 2002), [55]. The agile approach to development is about agility of the development process, development teams and their environment (Boehm & Turner, 2004), [39]. This approach incorporates shared ideals of various stakeholders, and a philosophy of regular providing the customers with product features in short time-frames (Southwell, 2002), [45]. This frequent and regular feature delivery is achieved by team based approach (Coram & Bohner, 2005), [47]. Agile teams consist of multi-skilled individuals **??**Fowler, 2002), [16]. The development teams also have on-site customers with substantial domain knowledge to help them better understand the requirements (Abrahamsson, Solo, Ronkainen, & Warsta, 2002), [37]. Multiple short development cycles also enable teams to accommodate request for change and provide the opportunity to discover emerging requirements (Highsmith, 2002), [55]. The agile approach promotes micro-project plans to help determine more accurate scheduling delivery commitments (Smits, 2006), [48].

M Lindvall, V Basili, B Boehm, P Costa, (2002), [17] summarize the working definition of agile methodologies as a group of software development processes that must be iterative (take several cycles to complete), incremental (not deliver the entire product at once), self-organizing (teams determine the best way to handle work), and emergent (processes, principles, and work structures are recognized during the project rather than predetermined). In the paper by (Abrahamsson, Warsta, Siponen & Ronkainen, 2003), in general, characterized agile software development by the following attributes: incremental, cooperative, straightforward, and adaptive [24]. Boehm, B., & Turner, R. (2005), generalize agile methods are lightweight processes that employ short iterative cycles, actively involve users to establish, prioritize, and verify requirements, and rely on a team's tacit knowledge as opposed to documentation [30].

1 II.

2 Agile Methods

For over a decade now, there has been an ever increasing variety of agile methods available includes a number of specific techniques and practices of software development. Agile methods are a subset of "iterative and evolutionary methods" [83,84] and are "based on iterative enhancement" [85] The major methods include eXtreme Programming (Beck, 1999), [82], Scrum (K. Schwaber & Beedle, 2002), [53], Dynamic Systems Development Method (Stapleton, 1997), Adaptive Software Development (Highsmith, 2000), Crystal **??**Cockburn, 2002), and Feature-Driven Development (Palmer & Felsing, 2002). [58], [59], [60], [61]. Figure 1 shows an agile software development methodology process flow (Scrum).

3 Year

The Agile Manifesto articulates the common principles and beliefs underlying these methods **??**Cockburn, 2002), [16]. Among the first and perhaps best known agile methods are Scrum and XP , [49]. See Figure 2 shows the current rate of Agile methodologies used. Scrum is aimed at providing an agile approach for managing software projects while increasing the probability of successful development of software, whereas XP focuses more on the project level activities of implementing software. Both approaches, however, embody the central principles of agile software development [31]. Agile software development processes –such as the Rational Unified Process (RUP), Extreme Programming (XP), Agile Unified Process (AUP), Scrum, Open Unified Process (OpenUP), and even Team Software Process (TSP) –are all iterative and incremental (evolutionary) in nature [63]. Some these modern approaches, in particular XP and Scrum, are agile in nature. The agile methods are focused on different aspects of the software development life cycle. Some focus on the practices (extreme programming, pragmatic programming, agile modeling), while others focus on managing the software projects (the scrum approach) [12].

4 III.

5 Comparison Agile Software Development Methodologies over Traditional SDMs

There are many different characteristics between ASDMs and TSDMs. Boehm [16], for example, reports nine agile and heavyweight discriminators. He believes the primary objective of ASDMs is on rapid value whereas the primary objective of TSDMs is on high assurance.

Study performed S. Nerur, R. Mahapatra, G. Mangalaraj [22] state a comparison of traditional and agile development, they report seven issues to differentiate traditional and agile development. Their fundamental assumption of traditional development: "system are fully specifiable, predictable and are built through meticulous and extensive planning", whereas agile development: "high-quality adaptive software is developed by small teams using the principles of continuous design improvement and testing based on rapid feedback and change".

T. Dyba, & T. Dingsoyr, [74] summarize the differences between Agile development and traditional development basis on the of an unpredictable world, as well as emphasizing the value competent people and their relationships bring to software development. Agile methods address the challenge of an unpredictable world, emphasizing the value competent people and their relationships bring to software development [74].

Different researchers compare traditional and agile approaches, in their different perspectives, are summarized in Table **??** (All sources from additional information). Linear; Life-cycle model (waterfall, spiral or some variation)

Iterative; The evolutionarydelivery model

Style of development , [50] Anticipatory Adaptive

Requirements (Boehm, 2002); (Boehm and Turner, 2004), [16], [39] Knowable early, largely stable; Clearly defined and documented

Emergent, rapid change, unknown -Discovered during the project Architecture (Boehm, 2002); **??**Wysocki, 2009(Wysocki, , 2011)) , [16], [56] Heavyweight

6 Predictability and optimization

7 Exploration or adaptation

Change , [19] Tend to be change averse Embrace change

Team members (Boehm, 2002) , (Sherehiy, Karwowski, & Layer, 2007), [16], [41] Distributed teams of specialists; Plan-oriented, adequate skills access to external knowledge Agile, knowledgeable, collocated and collaborative; Co-location of generalist senior technical staff;

Team organization , [52] Pre-structured teams Self-organizing teams Client Involvement), [21] Low involvement; Passive Client onsite and considered as a team member; Active/proactive Organization culture (Highsmith, 2002) , (Nerur, Mahapatra, Mangalaraj, 2005), [55], [22] Command and Control Culture

8 Leadership and Collaboration Culture

Software development process (Salo, & Abrahamsson, 2007), [42] Universal approach and solution to provide predictability and high assurance Flexible approach adapted with collective understanding of contextual needs to provide faster development Measure of success **??**Highsmith, 2010), **??**1] Conformance to plan Business value delivered a) Major agile benefits in comparison to the traditional approach

In this section, we presenting list and explain some of agile benefits in comparison to the traditional approach which significantly improves software development in many ways. We try to provide an indepth understanding (in some cases with figures), of these merit issues: Dagnino, 2002), they believe, Agile methods are iterative, evolutionary, and incremental delivery model of software development [30], [79], [29], [20], [80], [24], [81].

Entire application is distributed in incremental units called as iteration. Development time of each iteration is small (couple of weeks), fixed and strictly adhered to. Each iteration is a mini increment of the functionality and is build on top of previous iteration. Agile software development of short iterative cycles offers an opportunity for rapid, visible and motivating software process improvement [75]. Traditional approaches to the data-oriented aspects of software development; however, tend to be serial, not evolutionary and certainly not agile, in nature. **??**005), generalize agile methods are lightweight processes that employ short iterative cycles, actively involve users to establish, prioritize, and verify requirements, and rely on a team's tacit knowledge as opposed to documentation [30]. G Perera, & MSD Fernando (2007), also describe Agile practice is a customer oriented, light-weight software development paradigm, best suited for small size development teams in projects under vague and changing requirements [65]. A number of agile software development methods such as extreme programming (XP), feature-driven development, crystal clear method, scrum, dynamic systems development, and adaptive software development, fall into this category [22]. Traditional Software Development Methods (TSDMs) including waterfall and spiral models are often called heavyweight development methods [26]. These methods involves extensive planning, predefine process phases, heavy documentation and long term design process. Lightweight methodologies put extreme emphasis on delivering working code or product while downplayning the importance of formal process and comprehensive documentation [23]. lifecycle based software development delivers the software only after entire completion of development process and before that clients have no clear idea and view of software to be developed. According to (Boehm & Turner, 2005), Fast cycles, frequent delivery: Scheduling many releases with short time spans between them forces implementation of only the highest priority functions, delivers value to the customer quickly, and speeds requirements emergence [30]. ASD methods are iterative and incremental development [4], and each successful completion of development iteration, it delivers software product increment to client, thus Agile software development is satisfying the customer through early and continuous delivery of the valuable software [66]. Traditional, emerged as a response to the inability of previous plandriven approaches to handle rapidly changing environments (Highsmith, 2002). As second principle of Agile Manifesto [1] –welcome changing requirements, even late in development?, all agile method(s) is well organized, accommodate to change

requirements. According to B. Boehm, (2002), organizations -are complex adaptive systems in which requirements are emergent rather than pre-specifiable? and agile approaches -are most applicable to turbulent, highchange environments? [16] In contrast, agile development framework allows both customers and developers to change the requirements throughout the project, but only the customers have the authority to approve, disapprove and prioritize the ever-changing requirements (Koch, 2005), [57]. In traditional SDMs it increases complexity for accepting changing requirements while developing, and also increases and delivery time, as well as cost to deliver software product. Agile requirements prioritization techniques to support and deal with frequent changes in priority lists which have been identified as success issue to accommodate over changes [73]. In traditional development, software product with all features will be delivered at a time only after completion of software project. Customers are actively involved, and get higher priority in agile approaches rather than any traditional approaches. There is face to face communication and continuous feedback from customer (product owner) always happen in agile approach.

Figure **??** : Active customer involvement in agile approach Customers appreciate active participation in projects as it allows them to control the project and development process is more visible to them, as well as, they are kept up to date [73]. This customer involvement mitigates one of the most consistent problems on software projects: "What they will accept at the end of the project differs from what they told us at the beginning". This interaction helps the customer to form a better vision of the emerging product. Along with the ability to visualize the functionality that is coming based on having seen what was built so far, the customers develop a better understanding of their own needs and the vocabulary to express it to the developers [9]. Agile projects require a meaningful client involvement in every part of the project to provide constant feedback in an open and honest way **??**Wysocki, 2009), [57]. This feedback is a key element of agile methodologies, which is why the customer must be committed, knowledgeable, collaborative, representative, and empowered to avoid risk of failure (Boehm, 2002), [16]. People are the primary drivers of agile projects and agile teams work best when people are physically close and document preparation and dissemination are largely replaced by face-to-face communication and collaboration , [21].

9 vii. Reduce cost and time

The study reports conducted by B. Bahli and ESA Zeid [77] that the development team found using the waterfall model to be an "unpleasant experience", while XP (an agile method) was found to be "beneficial and a good move from management". The XP project was delivered a bit less late (50% time-overrun, versus 60% for the traditional), and at a significantly reduced cost overrun (25%, compared to 50% cost overrun for the traditional project). Agile development involves less cost of development as rework, management, documentation and other non-development work related cost is reduced. Figure 11 : Design phase composition between waterfall and agile development According to (Boehm & Turner, 2005), agile approach design is simple which involves Designing for the battle, not the war. The motto is YAGNI (You Aren't Going to Need It). The antimotto is BDUF (Big Design Up Front). Strip designs down to cover just what you're developing. Since change is inevitable, planning for future functions is a waste of effort [30]. Customer gets to know regular and frequent status of the application and delivery is defined by fixed timescale. So, customer is assured of receiving some functionality by a fixed time period. Due to the short development life cycle through an iterative and incremental process, the agile methods have been used widely in business sectors where requirements are relatively unstable [26]. ix. Self organized team Agile teams are self organizing and roles and relationships evolve as necessary to meet objectives . Team composition in an agile project is usually cross-functional and self-organizing, without consideration for any existing corporate hierarchy or the corporate roles of team members **??**4]. Agile product development practices introduce changes in team culture in an attempt to bringing reciprocal effects of roalty and commitment to the team and projects (Sherehiy, Karwowski, & Layer, 2007). Team members normally take responsibility for tasks that deliver the functionality an iteration requires. They decide individually how to meet an iteration's requirements. Teams develop applications collaboratively and in cooperative environment. Agile alliance [5], claims that for a given problem size, "fewer people are needed if a lighter methodology is used, and more people are needed if a heavier methodology is used," and asserts that, "There is a limit to the size of problem that can be solved with a given number of people" [44]. According to (Boehm & Turner, 2005), agile approach design is simple which involves Designing for the battle, not the war. The motto is YAGNI (You Aren't Going to Need It). The anti-motto is BDUF (Big Design Up Front). Strip designs down to cover just what you're developing. Since change is inevitable, planning for future functions is a waste of effort [30]. In their research paper [46], (K Molokken & Ostvold, 2005), define agile method(s) as a flexible software development model(s), basis on evolutionary and incremental models; and also claim that, among the benefits of using these models are reduced software project overruns.

xii. Improves Software Quality Boehm, B., & Turner, R. (2004, May), Agile development methodologies (such as XP, Scrum, and ASD) promise higher customer satisfaction, lower defect rates, faster development times and a solution to rapidly changing requirements. Plan-driven approaches such as Cleanroom, the Personal Software Process, or methods based on the Capability Maturity Model promise predictability, stability, and high assurance [38].

The regular and continuous interaction between the customer and the developers have as their primary objective assuring that the product as built does what the customer needs for it to do and assures the usability

of the product as well. The strong technical focus results in much better testing on an Agile project than in most other methods **??**9]. According to Charvat, (2003), agile practices: iterative and adaptive life cycles have the advantage of a continual testing throughout the project, which has a positive impact on quality [43]. Agile developers take responsibility for the quality of the code they write. In addition to producing cleaner code, it means that if there are testing specialists on the project, they will start their testing with better software, which always results in more effective testing and a better resulting product. In addition to, developers value the technical focus on testing and refactoring of agile methods increasing their motivation. There is also a perception of increased quality in software products and higher productivity when using some agile teams use practices like coding standards, peer reviews, and pair programming to assure that the code they produce is technically solid [73].

xiii. Increase business value, visibility, adaptability and reduce cost Agile software development accelerates the delivery of initial business value, and through a process of continuous planning and feedback, ensures that value continues to be maximized throughout the development process. ASD provides customer satisfaction through collaboration and frequent delivery of implemented features. By delivering working, tested, deployable software on an incremental basis, agile development delivers increased value, visibility and adaptability much earlier in the life cycle, significantly reducing project risk. In a study by Boehm and Papaccio [72] discovered that a typical project experiences a 25% change in requirements, while yet another **??**Johnson] showed that 45% of features were never used. Agile approach aims to reduce waste and over-production by determining which parts are actually needed by the customer at each stage. In Agile approaches, delivering software on an incremental basis, customers give continuous feedback and agile team will always deliver products on time and on budget. As traditional project management isn't succeeding, more and more companies are turning to Agile development. According to the Standish Group's, **??**11] famous CHAOS Report of 2000, 25% of all projects fail outright through eventual cancellation, with no useful software deployed. Sadly, this represents a big improvement over CHAOS reports from past years. Recently, they conduct a survey for Agile implementation success rate, see figure 19.

10 Agile Adoption

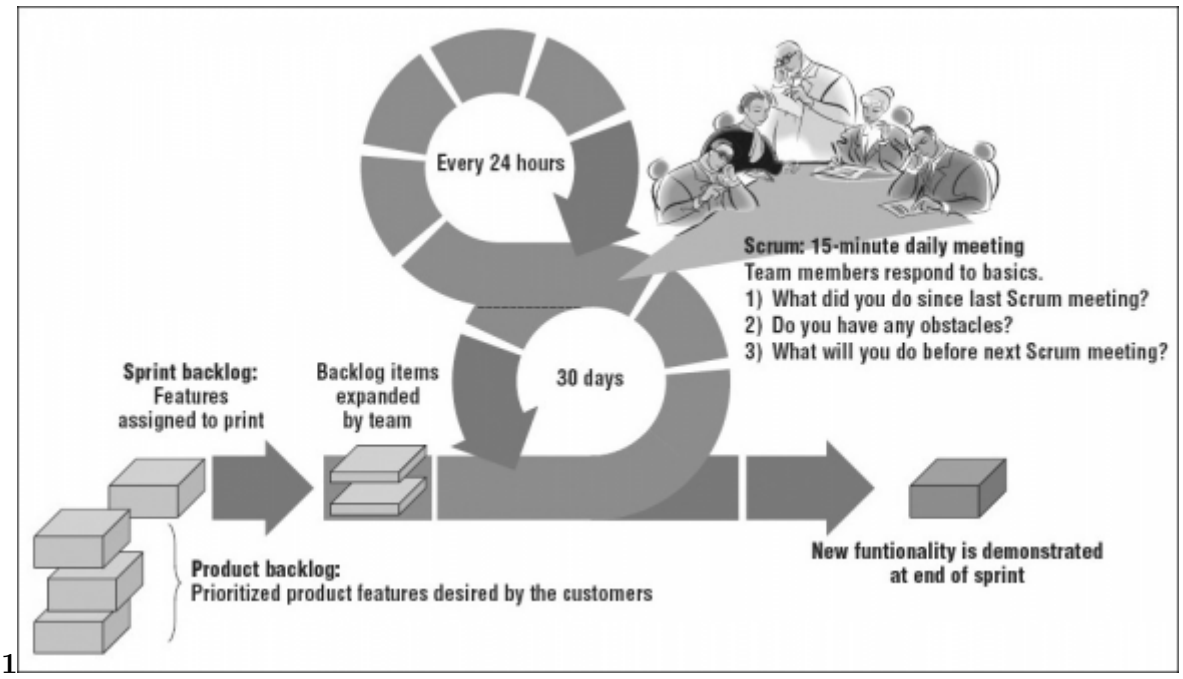
Agile methods are highly being adopted because of expectations that these methods can bring development success (Esfahani, Yu, & Annosi, 2010). One of the main reasons for success with agile methods is that they are highly adaptive , [38]. Figure 1 reveals the current levels of agile adoption. In this case, 71% of respondents indicated that they work in organizations that have succeeded at agile and an additional 15% work in organizations that have tried agile but have not yet succeed at it. According to (West & Grant, 2010), "in the past few years, Agile processes have not only gained increasing adoption levels; they have also rapidly joined the mainstream of development approaches" [28]. Mary large companies including HP, IBM, Oracle, and Microsoft use Agile methodologies [76] -and more and more smaller organisations turn Agile each year. In their study (West & Grant, 2010), conducted by Forrester Research in 2009, agile software development processes were in use in 35% of organizations, and another 16% of organizations used an iterative development approach, while only 13% of organization use a Waterfall approach. However, nearly 31% did not use a formal development methodology [28]. The main reasons behind for adopting Agile approaches rather than plan-driven approaches relate to: rapid changes; need for rapid results; emergent requirements , [38]. According to Charvat, (2003), , & Perrin, (2008), Agile methodologies have numerous advantages including that they: adapt very well to change and dynamism; are people-oriented and value-driven, rather than process-oriented and plan-driven; mitigate risks by demonstrating values and functionalities up front in the development process; provide a faster time to market; improve productivity (by reducing the amount of documentation) and will fail early/quickly and painlessly, if a project is not doable [34], [33], [32].

A state of Agile survey 2011, conducted by versionone Inc. result shows: the top three reasons for adopting Agile to -accelerate time to market, increase productivity, and to more easily manage changing priorities. Prior to adoption, respondents said productivity and time to market ranked as their top reasons to adopt agile. But experienced agile users said actual benefits were primarily project visibility (77%) and the ability to manage changing priorities (84%). 5. Conclusion Agile software development methodologies are evolutionary and incremental models have become increasingly popular in software development industry. Through, in many organizations, agile system development methods at adoption stage, agile methods might start to become well-established processes of these small, mid-level, even large organizations. There is increasing need to have a deeper understanding of agile methods in use in software development industry; as well as, have a better understanding -the benefits of agile approach as for accepting agile methods into their development style and for cope-up with their dynamic business needs. In this paper, we present main issues of agile numerous benefits in comparison to the traditional approach which significantly improves software development process in many ways. We also provide with this paper, the current adoption state of Agile software development with different current survey results with graphs. The purpose of this paper is to provide an in-depth understandingthe benefits of agile development approach into the software development industry, as well as provide a comparison study report of ASDM over TSDM. [1] [2]

[1] © 2013 Global Journals Inc. (US) Global Journal of Computer Science and Technology

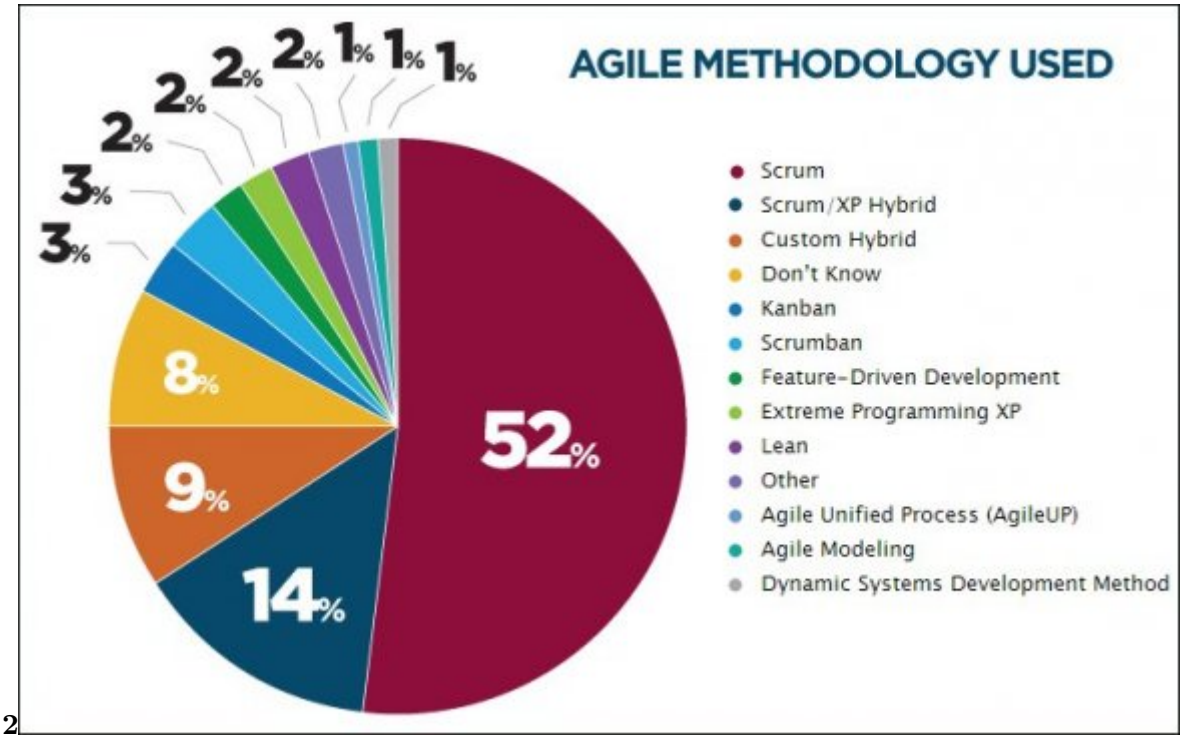
[2] © 2013 Global Journals Inc. (US)

Figure 1:



1

Figure 2: Figure 1 :



2

Figure 3: Figure 2 :

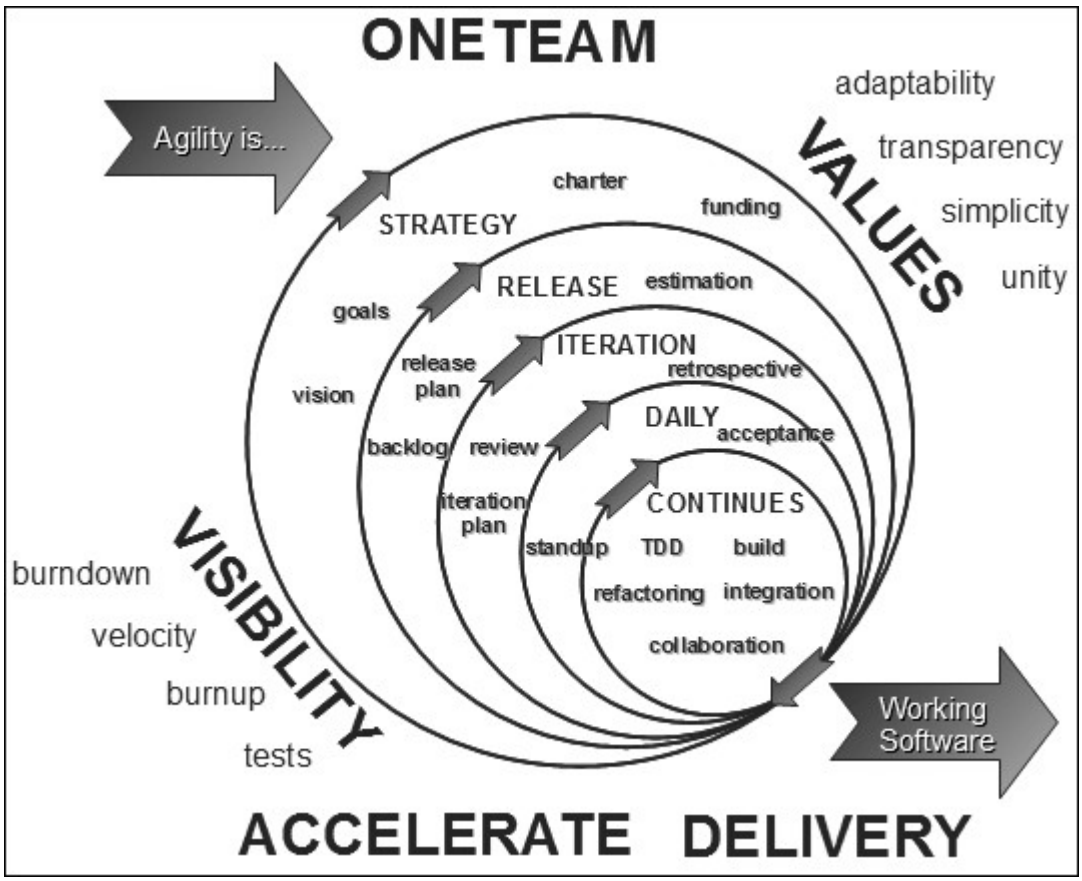
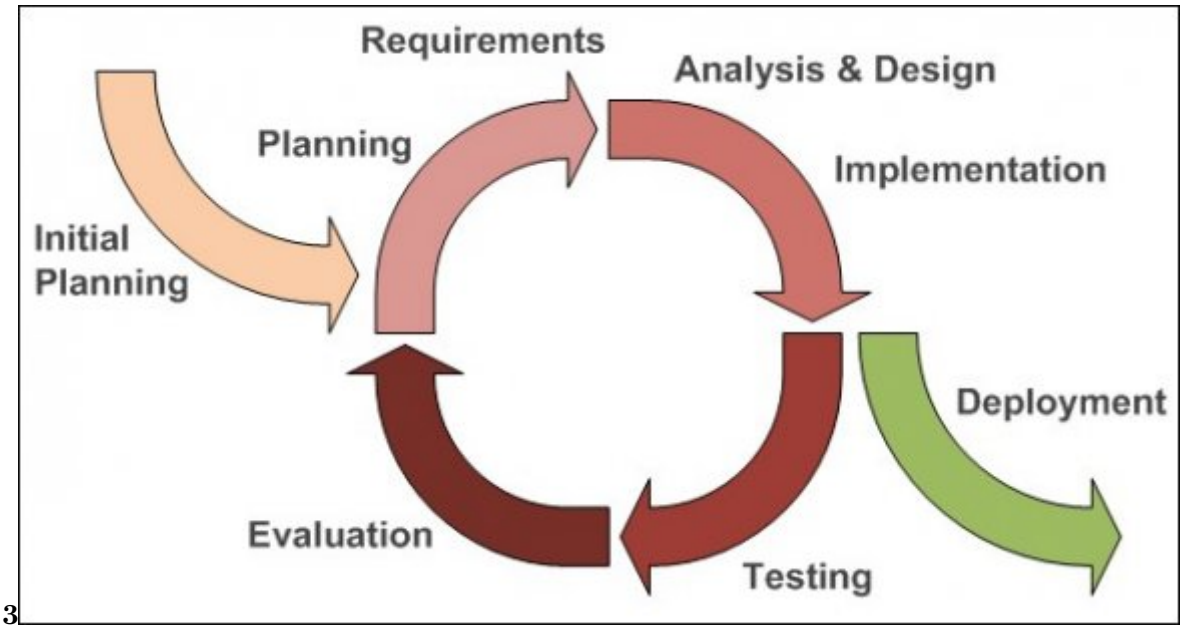
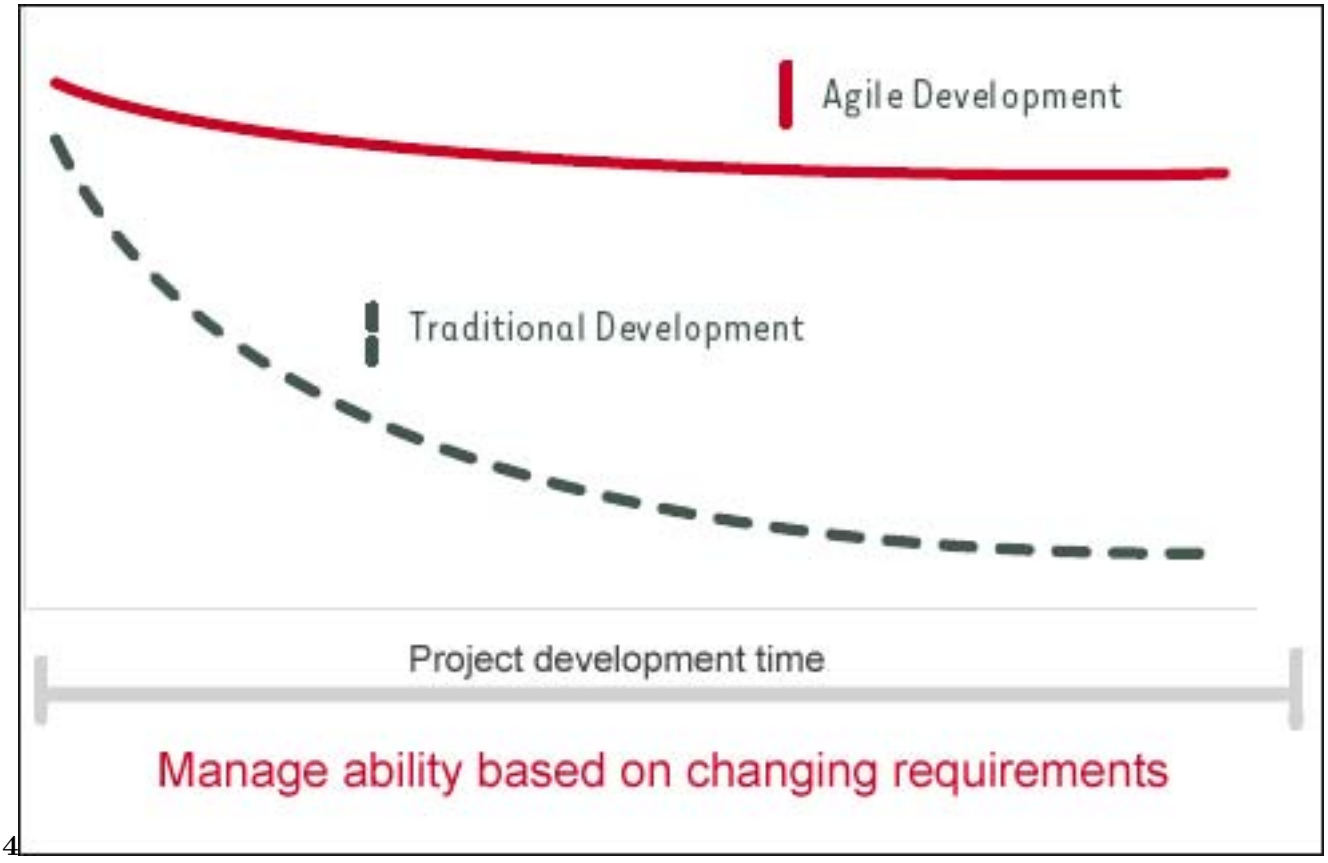


Figure 4:



3

Figure 5: Figure 3 :



4

Figure 6: Figure 4 :

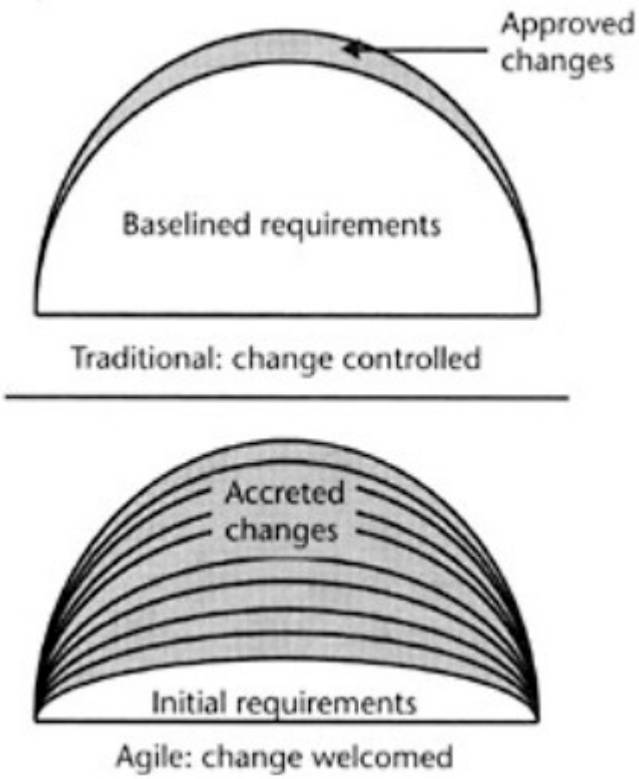
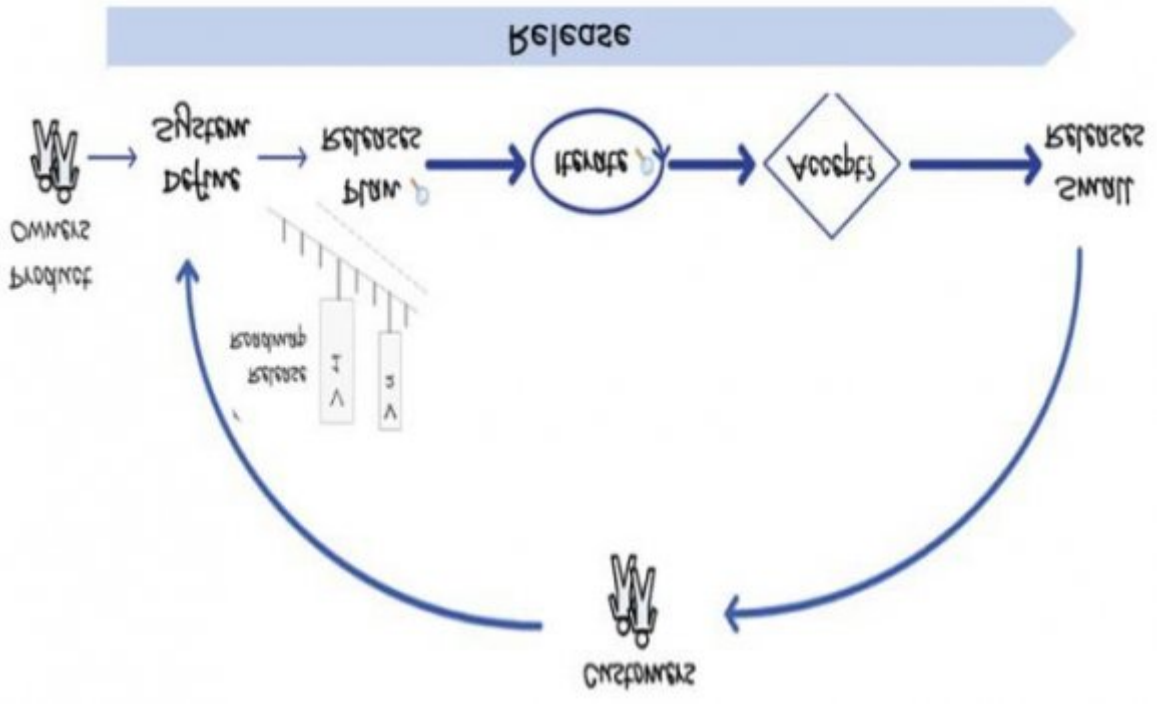
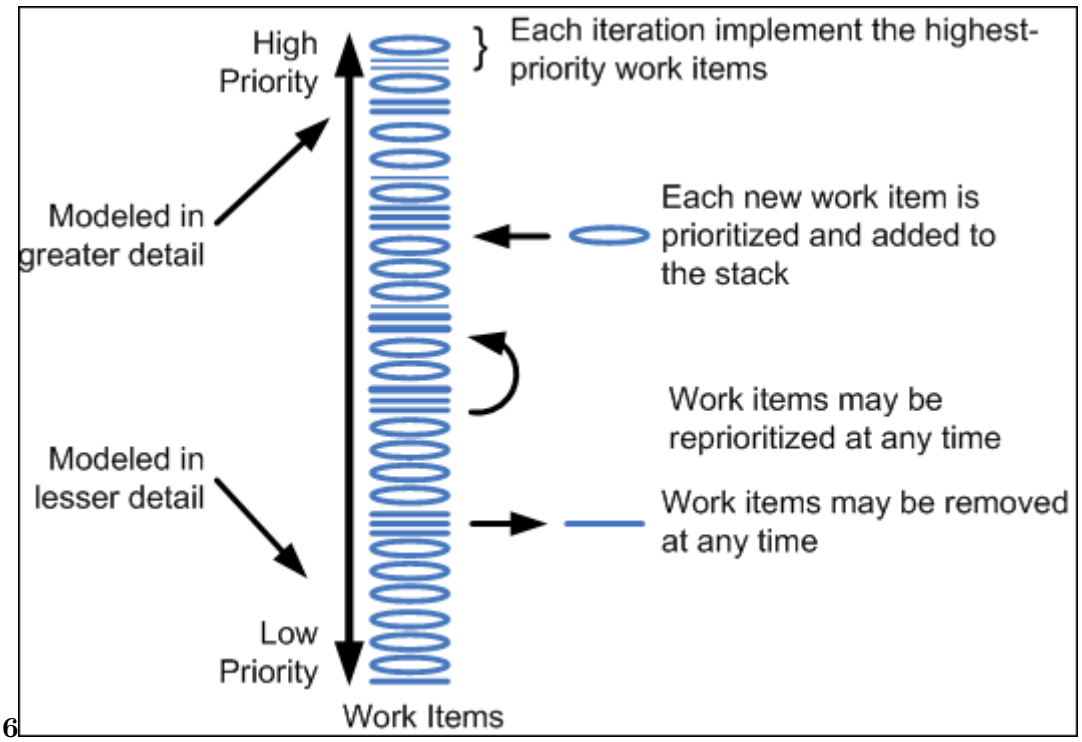


Figure 7:



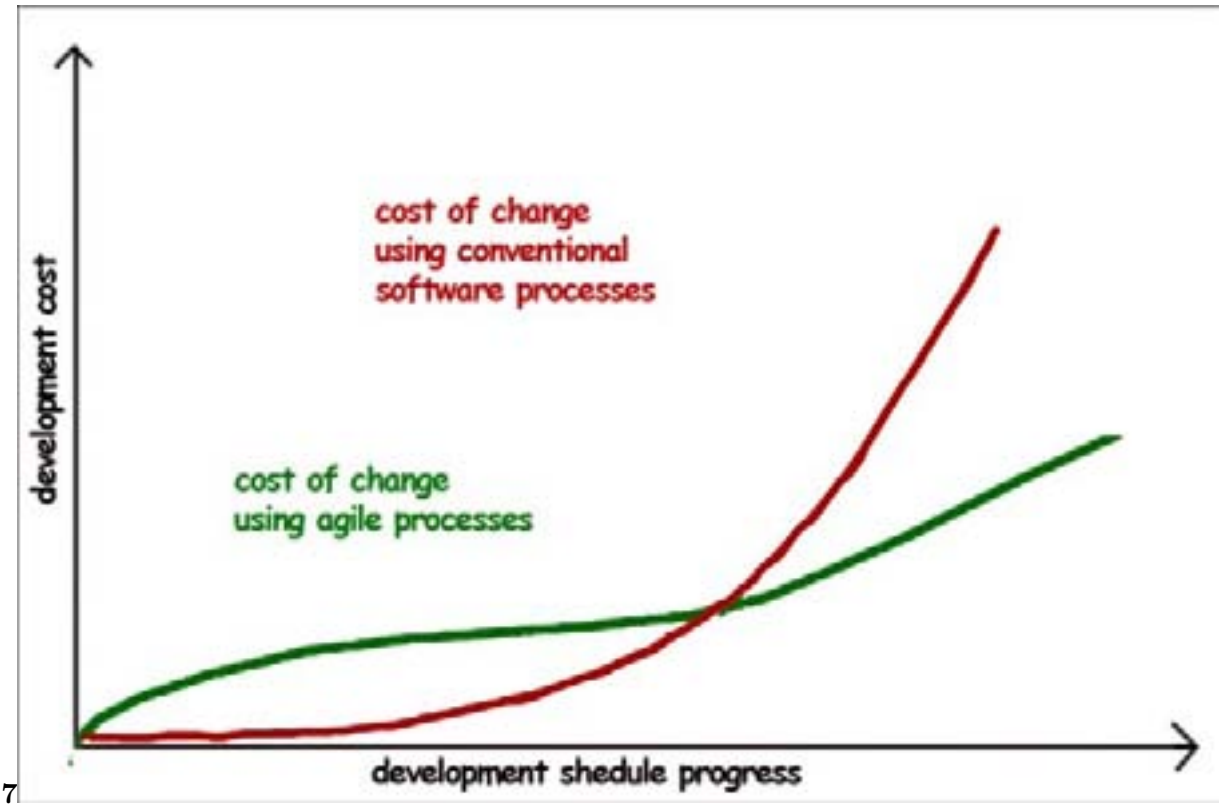
5

Figure 8: Figure 5 :



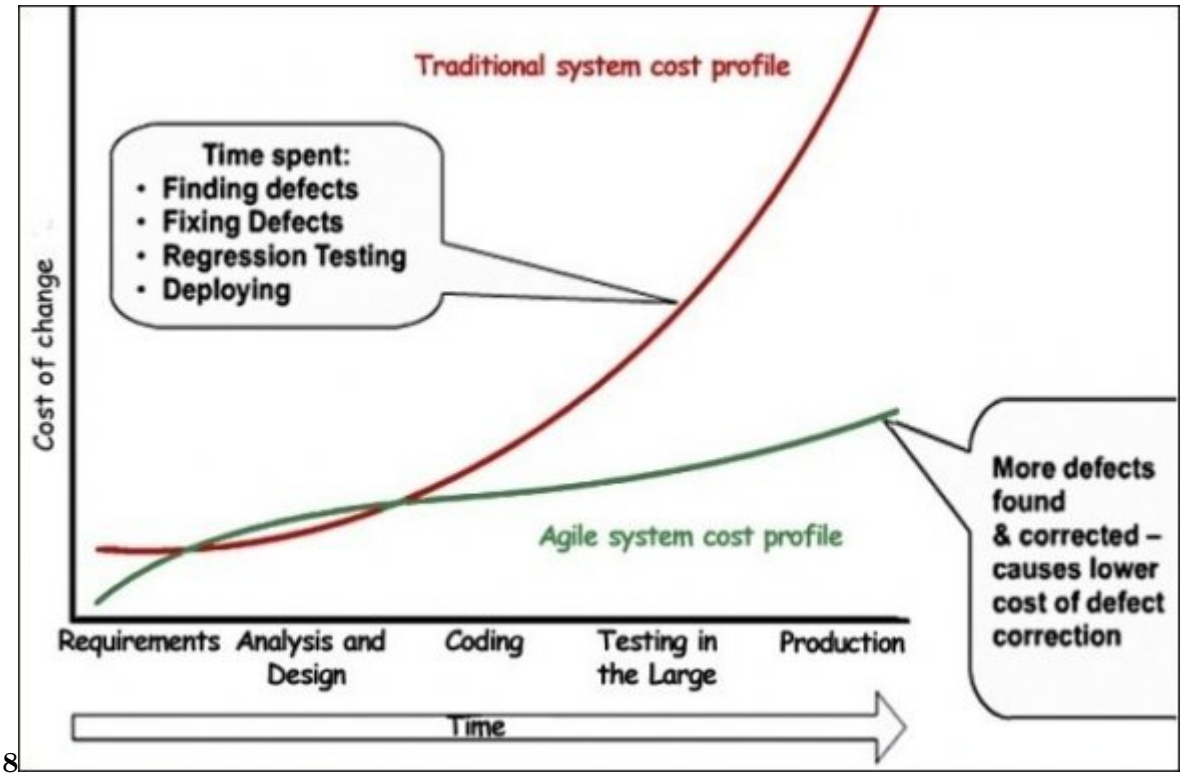
6

Figure 9: Figure 6 :



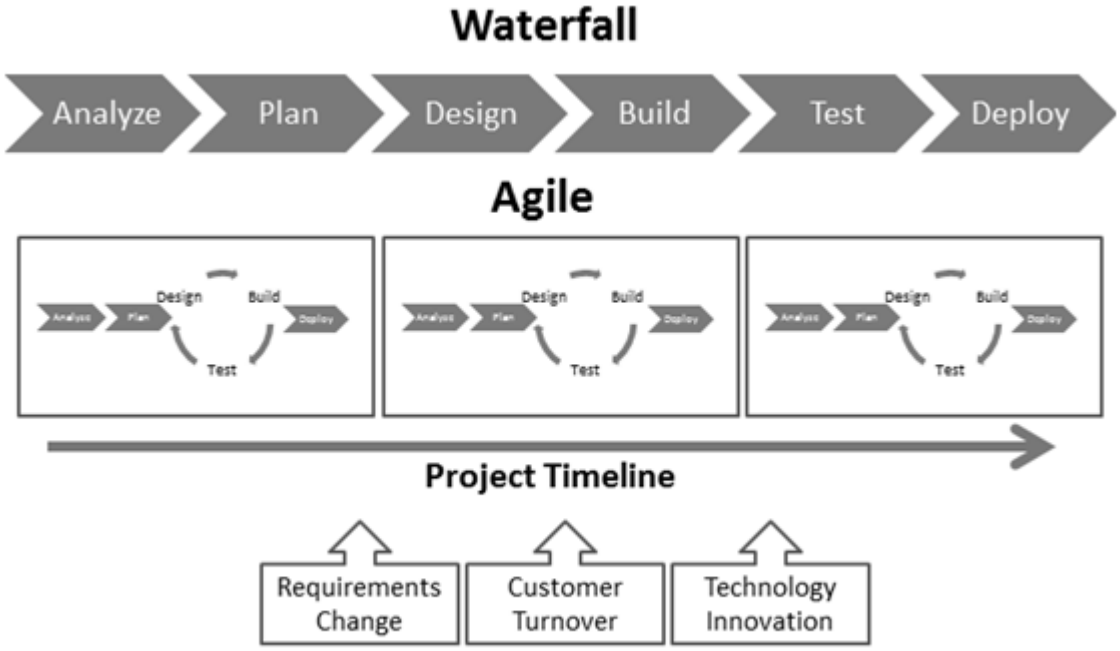
7

Figure 10: Figure 7 :C



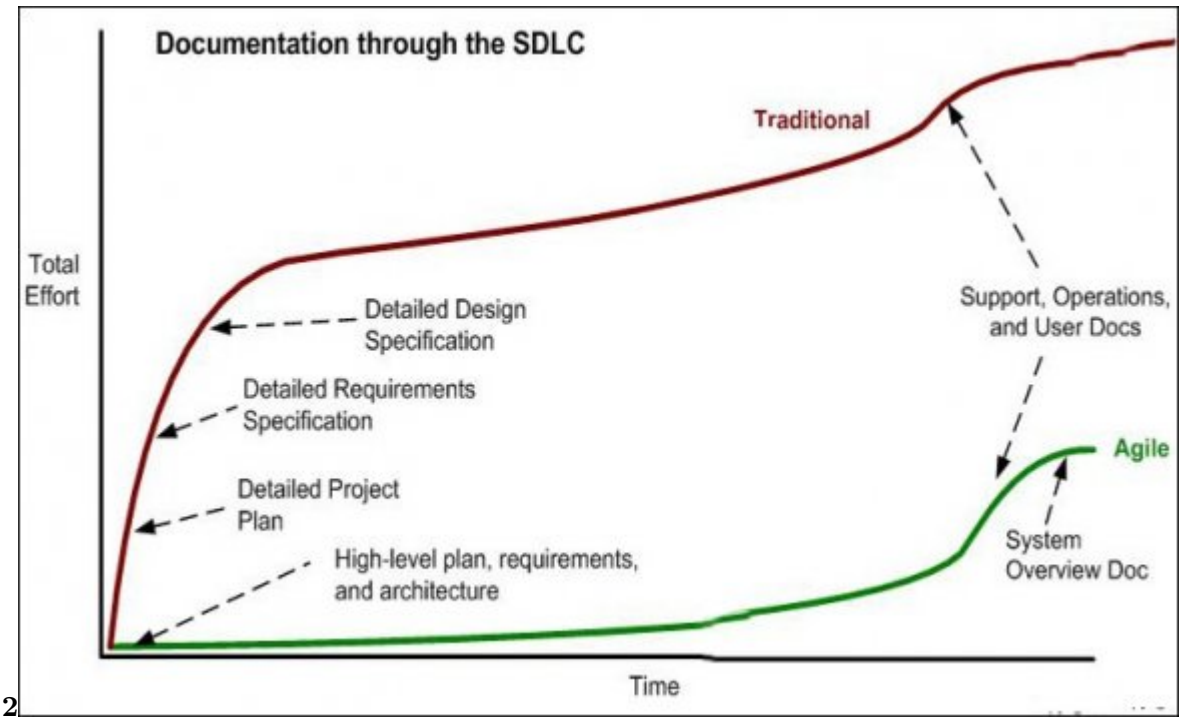
8

Figure 11: Figure 8 :



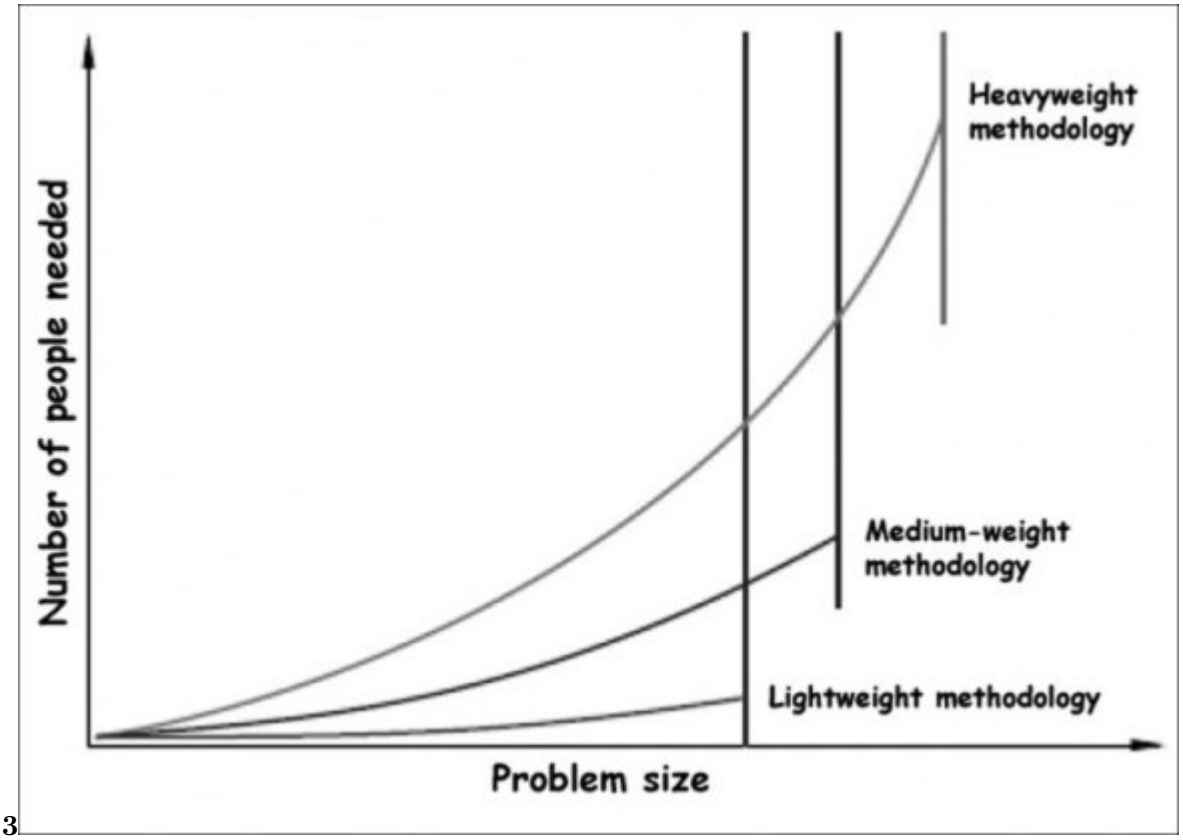
10

Figure 12: Figure 10 :



12

Figure 13: Figure 12 :C



13

Figure 14: Figure 13 :

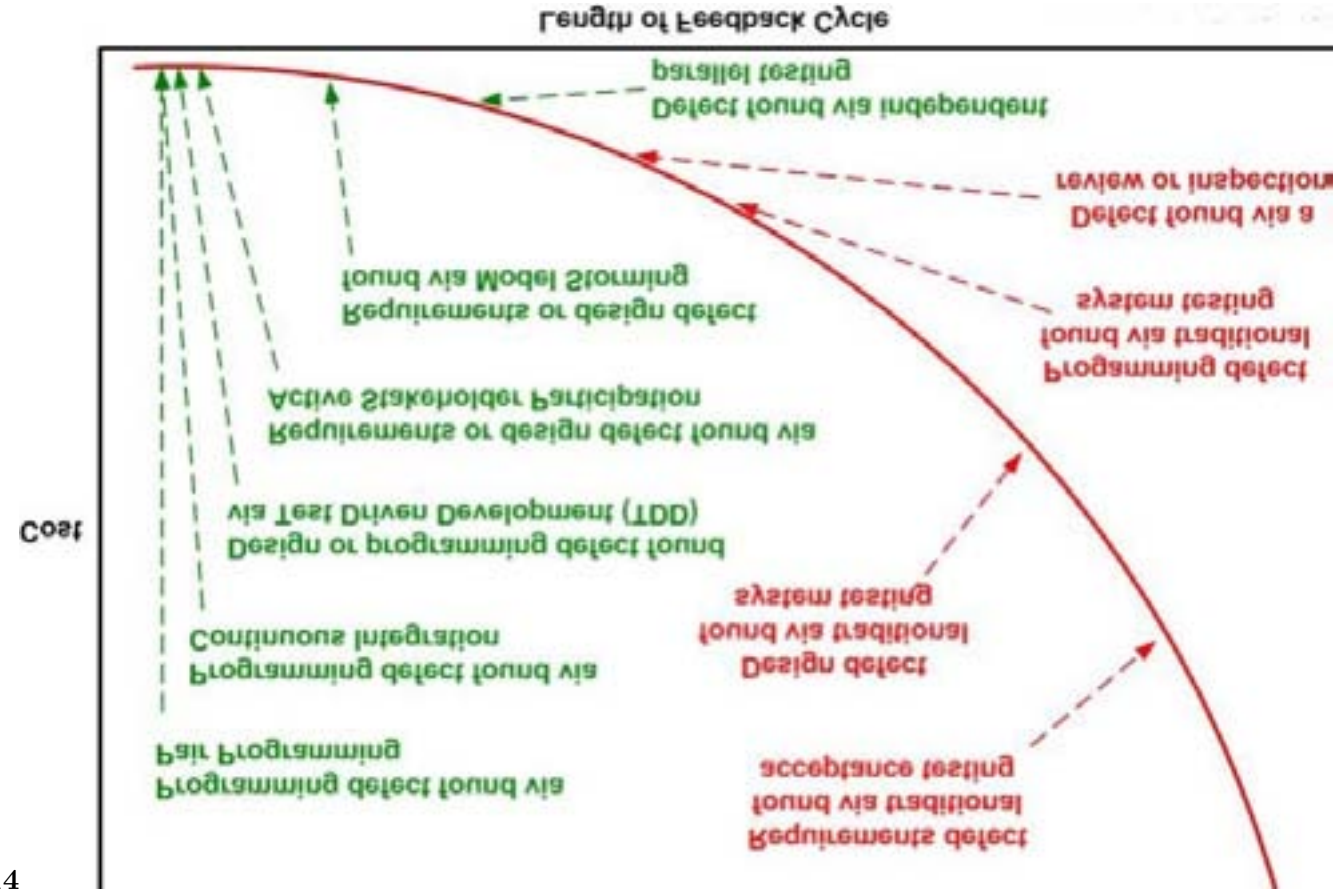
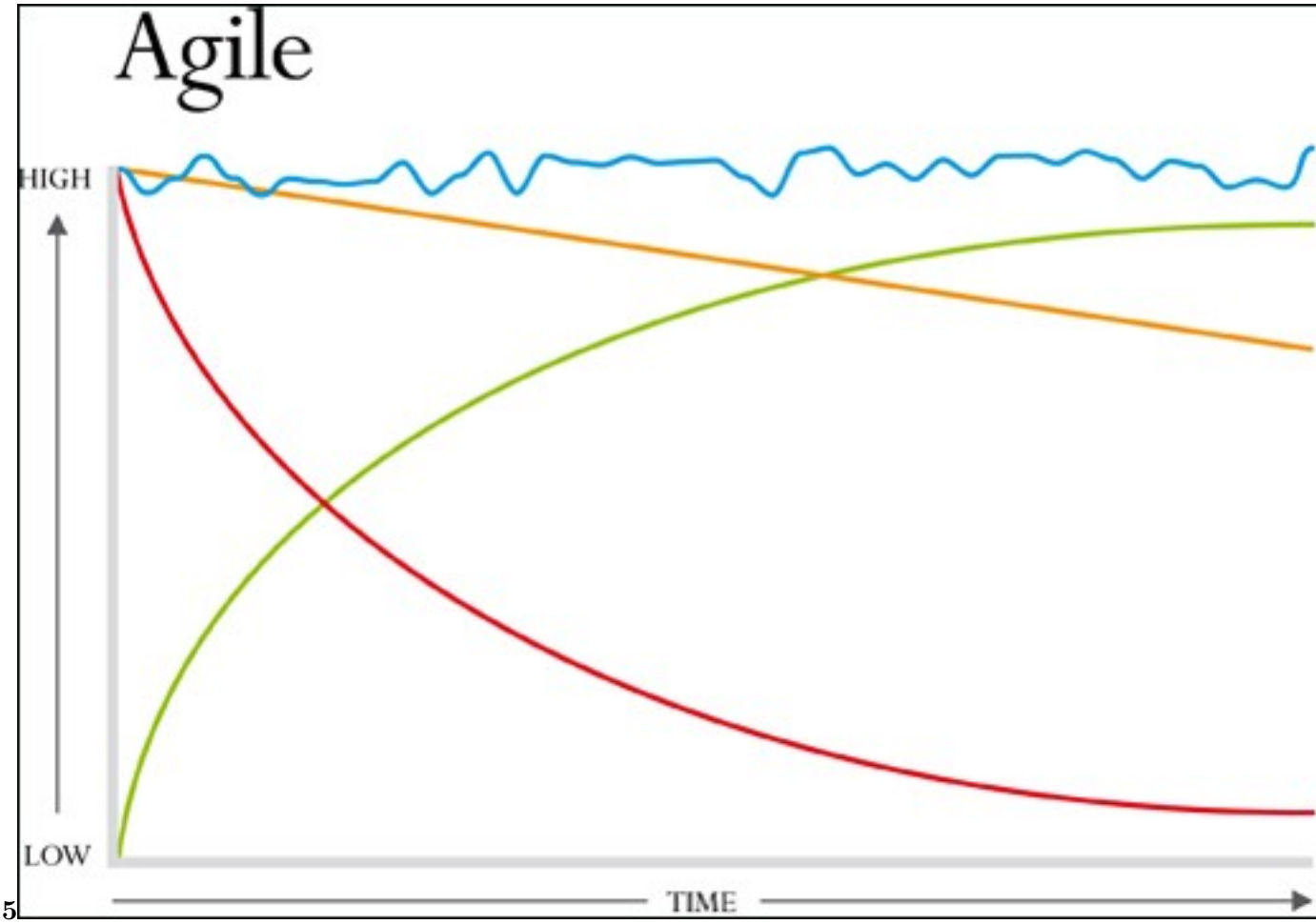
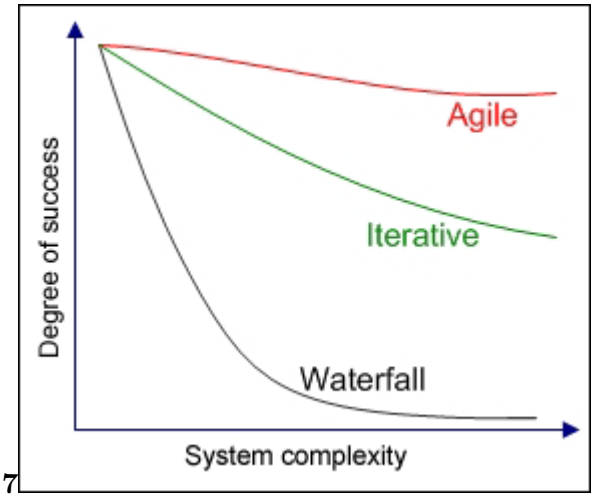


Figure 15: Figure 14 :



15

Figure 16: Figure 15 :



17

Figure 17: Figure 17 :

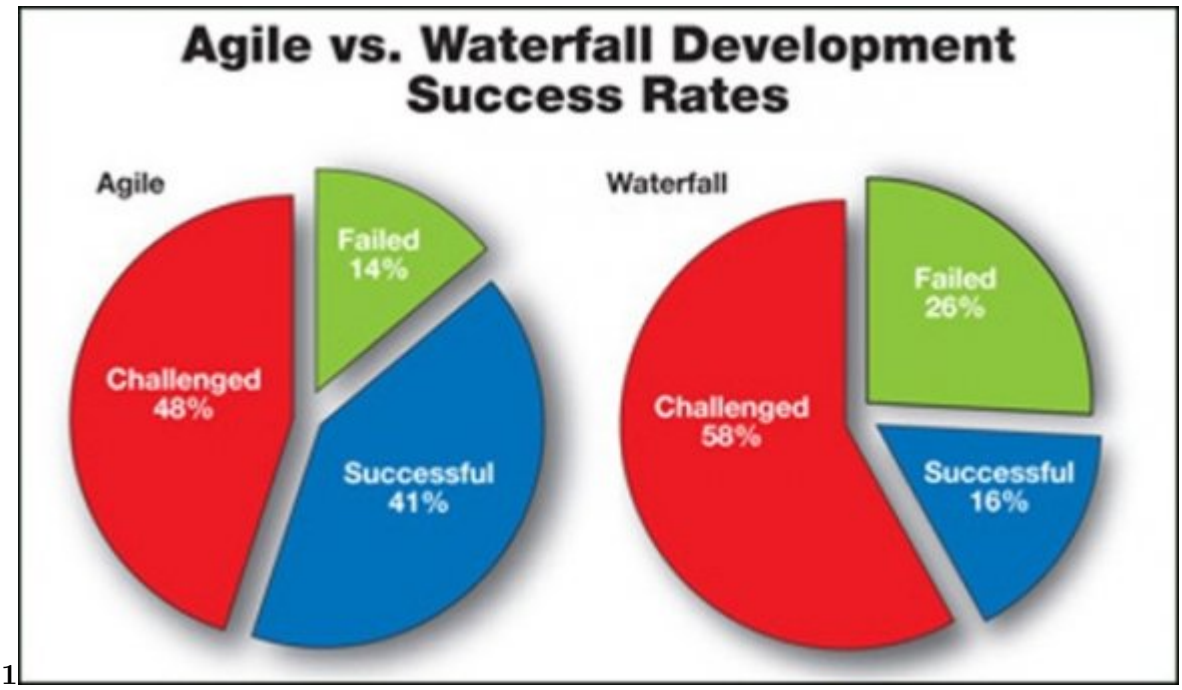


Figure 18: Figure 19 :Figure 21 :

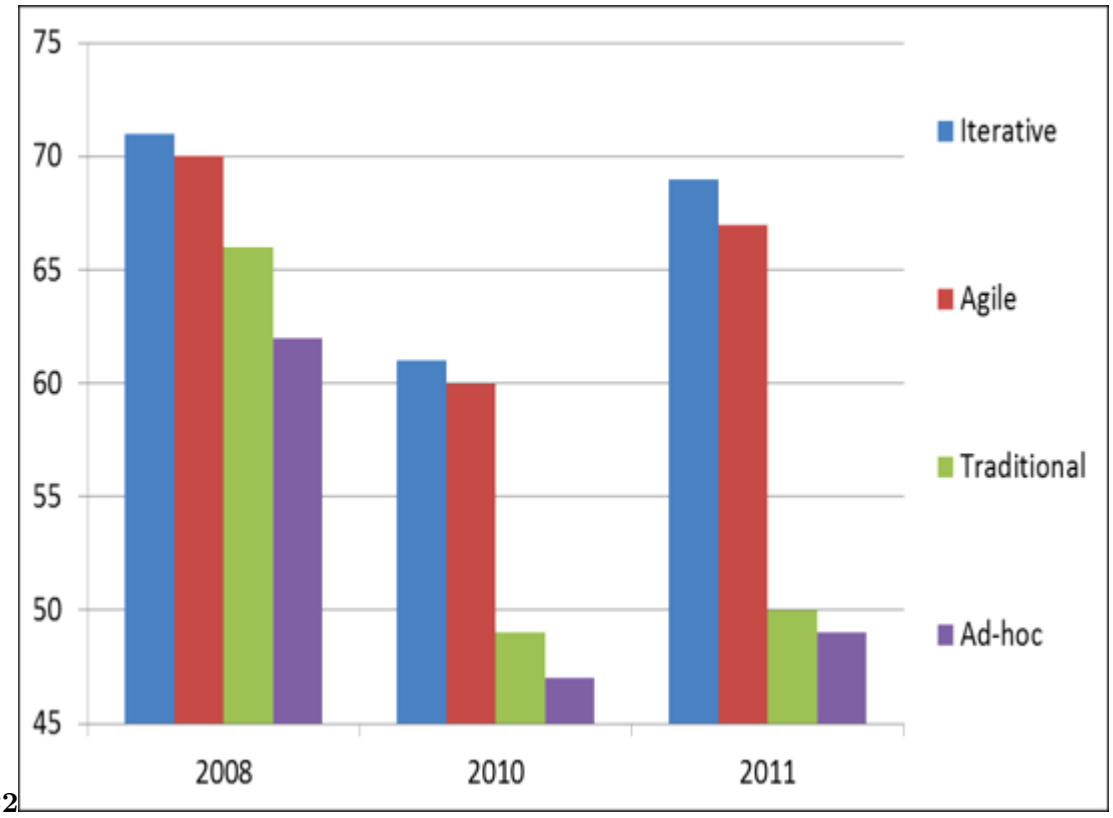


Figure 19: Figure 22 :

2023

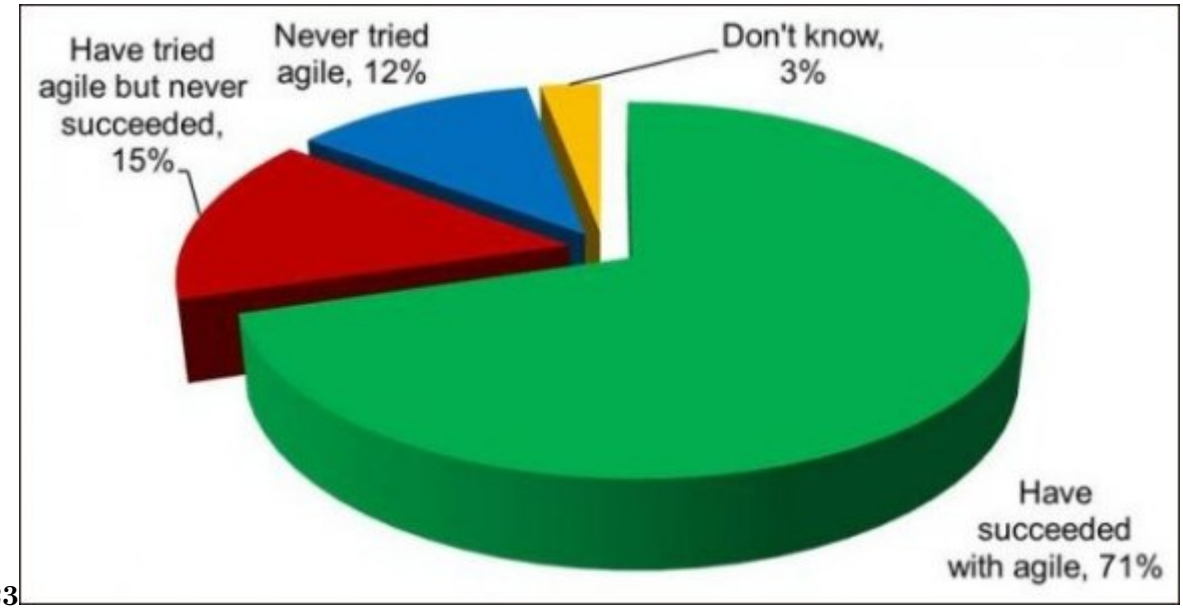


Figure 20: Figure 20 :Figure 23 :

24

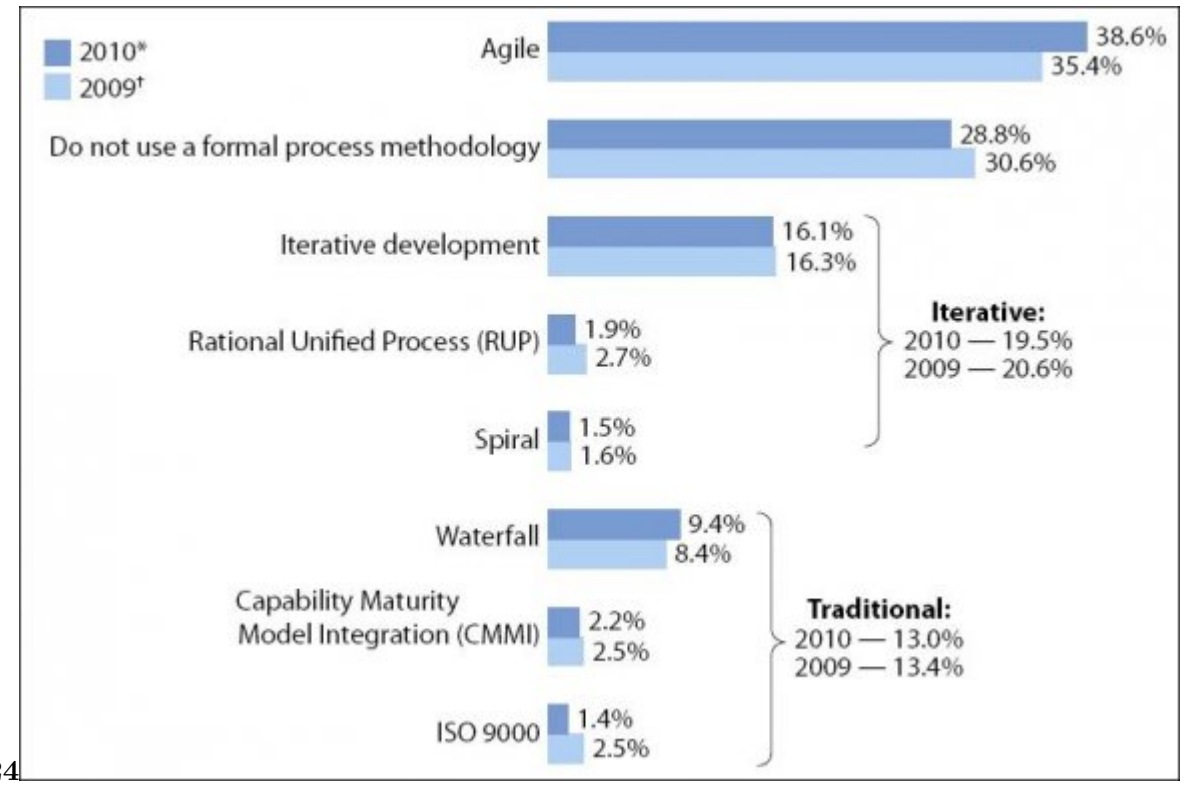


Figure 21: Figure 24 :

[Information Systems Management] , *Information Systems Management* 23 (3) p. .

[Cockburn] , A Cockburn . http://alistair.cockburn.us/crystal/articles/ms/methodologyspace.htmAccessedon2/2/2005 *The Methodology Space?*

[Petersen and Wohlin ()] 'A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case'. K Petersen , C Wohlin . *Journal of Systems and Software* 2009. 82 (9) p. .

[Molokken-Ostvold and Jorgensen ()] 'A comparison of software project overruns-flexible versus sequential development models. Software Engineering'. K Molokken-Ostvold , M Jorgensen . *IEEE Transactions on* 2005. 31 (9) p. .

[Larman and Basili (2003)] 'A History of Iterative and Incremental Development'. C Larman , V Basili . *IEEE Computer* June 2003. 36 (6) p. .

[Palmer and Felsing ()] *A Practical Guide to Feature Driven Development*, S Palmer , Felsing . 2002. Prentice Hall.

[Sherehiy et al. ()] 'A review of enterprise agility: Concepts, frameworks, and attributes'. B Sherehiy , W Karwowski , J K Layer . *International Journal of Industrial Ergonomics* 2007. 37 (5) p. .

[Highsmith ()] *Adaptive software development*, J A Highsmith . 2000. Dorset House.

[Larman ()] *Agile and iterative development: a manager's guide*, C Larman . 2004. Addison-Wesley Professional.

[Larman ()] *Agile and Iterative Development: A Manager's Guide*, C Larman . 2004. Boston: Addison Wesley.

[West et al. ()] *Agile development: Mainstream adoption has changed agility*, D West , T Grant , M Gerush , D Silva . 2010. Forrester Research.

[Fowler and Highsmith ()] 'Agile methodologists agree on something'. M Fowler , J Highsmith . *Software Development* 2001. 9 p. .

[Salo and Abrahamsson ()] 'Agile methods in European embedded software development organisations: a survey on the actual use and usefulness of Extreme Programming and Scrum'. O Salo , P Abrahamsson . *Software* 2008. 2 (1) p. . (IET)

[Salo and Abrahamsson ()] 'Agile methods in European embedded software development organisations: a survey on the actual use and usefulness of Extreme Programming and Scrum'. O Salo , P Abrahamsson . *Software* 2008. 2 (1) p. . (IET)

[Southwell ()] 'Agile process improvement'. K Southwell . *TickIT International Journal* 2002. p. .

[Garg ()] *Agile Software Development*, A Garg . 2009.

[Highsmith ()] *Agile software development ecosystems*, J A Highsmith . 2002. Addison-Wesley Professional.

[Abrahamsson et al. ()] *Agile Software Development Methods*, P Abrahamsson , O Solo , J Ronkainen , J Warsta . 2002. VTT technical Research Centre of Finland

[Cockburn and Highsmith ()] 'Agile software development, the people factor'. A Cockburn , J Highsmith . *Computer* 2001. 34 (11) p. .

[Koch ()] *Agile software development: evaluating the methods for your organization*, A S Koch . 2005. (Artech house)

[Williams and Cockburn ()] 'Agile software development: it's about feedback and change'. L Williams , A Cockburn . *IEEE Computer* 2003. 36 (6) p. .

[Williams and Cockburn (2003)] 'Agile Software Development: It's about Feedback and Change'. L Williams , A Cockburn . *IEEE Computer* June 2003. p. .

[Highsmith and Cockburn ()] 'Agile software development: The business of innovation'. J Highsmith , A Cockburn . *Computer* 2001. 34 (9) p. .

[Highsmith and Cockburn ()] 'Agile software development: The business of innovation'. J Highsmith , A Cockburn . *Computer* 2001. 34 (9) p. .

[Leffingwell ()] *Agile software requirements: Lean requirements practices for teams, programs, and the enterprise*, D Leffingwell . 2007. Addison-Wesley Professional.

[Glass (2001)] 'Agile Versus Traditional: Make Love, Not War'. R Glass . *Cutter IT Journal* Dec. 2001. p. .

[Beck and Boehm ()] 'Agility through discipline: A debate'. K Beck , B Boehm . *Computer* 2003. 36 (6) p. .

[Dagnino (2002)] 'An evolutionary lifecycle model with Agile practices for software development at ABB. In Engineering of Complex Computer Systems'. A Dagnino . *Eighth IEEE International Conference on*, 2002. December. 2002. IEEE. p. .

[Salo and Abrahamsson ()] 'An iterative improvement process for agile software development'. O Salo , P Abrahamsson . *Software Process: Improvement and Practice*, 2007. 12 p. .

[Turk et al. ()] 'Assumptions underlying agile software-development processes'. D Turk , F Robert , B Rumpe . *Journal of Database Management (JDM)* 2005. 16 (4) p. .

[Boehm and Turner (2004)] 'Balancing agility and discipline: Evaluating and integrating agile and plan-driven methods'. B Boehm , R Turner . *Proceedings. 26th International Conference on*, (26th International Conference on) 2004. May. 2004. 2004. IEEE. p. . (Software Engineering)

[Boehm et al. ()] 'Balancing plan-driven and agile methods in software engineering project courses'. B Boehm , D Port , A W Brown . *Computer Science Education* 2002. 12 (3) p. .

[Scanlon-Thomas ()] *Breaking the Addiction to Process: An Introduction to Agile Project Management*, E Scanlon-Thomas . 2011. (Itgp)

[Vinekar et al. ()] 'Can agile and traditional systems development approaches coexist? An ambidextrous view'. V Vinekar , C W Slinkman , S Nerur . *Information systems management* 2006. 23 (3) p. .

[Nerur et al. ()] 'Challenges of migrating to agile methodologies'. S Nerur , R Mahapatra , G Mangalaraj . *Project management methodologies. Selecting, implementig, and supporting*, 2005. 2003. 48 p. . (Charvat, J.)

[Cockburn ()] *Crystal clear: a humanpowered methodology for small teams*, A Cockburn . 2005. Addison-Wesley Professional.

[Curtis ()] B Curtis . *Three Problems Overcome with Behavioral Models of the Software Development Process (Panel)," International Conference on Software Engineering*, (Pittsburgh, PA) 1989. p. .

[Stapleton ()] *DSDM, dynamic systems development method: the method in practice*, J Stapleton . 1997. Addison-Wesley Professional.

[Dybå and Dingsøyr ()] T Dybå , T Dingsøyr . *Empirical studies of agile software development: A systematic review. Information and software technology*, 2008. 50 p. .

[Wysocki ()] *Effective project management: traditional, agile, extreme*, R K Wysocki . 2011. Wiley.

[Beck ()] 'Embracing change with extreme programming'. K Beck . *Computer* 1999. 32 (10) p. .

[Lindvall et al. ()] *Empirical findings in agile methods*, M Lindvall , V Basili , B Boehm , P Costa , K Dangle , F Shull , . . Zelkowitz , M . 2002. 2002. p. .

[Perera and Fernando (2007)] 'Enhanced agile software developmenthybrid paradigm with LEAN practice. In Industrial and Information Systems'. G I U S Perera , M S D Fernando . *ICIIS 2007. International Conference on*, 2007. August. 2007. IEEE. p. .

[Livermore ()] 'Factors that significantly impact the implementation of an agile software development methodology'. J A Livermore . *Journal of Software* 2008. 3 (4) p. .

[Boehm ()] 'Get ready for agile methods, with care'. B Boehm . *Computer* 2002. 35 (1) p. .

[Reifer ()] 'How good are agile methods'. D J Reifer . *Software*, 2002. IEEE. 19 p. .

[Lemétayer ()] *identifying the critical factors in software development methodology FIT*, J Lemétayer . 2010.

[Cho ()] 'Issues and Challenges of Agile Software Development with Scrum'. J Cho . *Issues in Information Systems* 2008. 9 (2) p. .

[Larman and Basili ()] 'Iterative and incremental developments. a brief history'. C Larman , V R Basili . *Computer* 2003. 36 (6) p. .

[Basili and Turner ()] 'Iterative Enhancement: A Practical Technique for Software Development'. V R Basili , A J Turner . *IEEE Transactions on Software Engineering* 1975. 1 (4) p. .

[Smits ()] *Levels of Agile Planning: From Enterprise Product Vision to Team Stand-up*, H Smits . 2006. (Rally Software Development Corporation Whitepaper)

[Boehm and Turner ()] *Management challenges to implementing agile processes in traditional development organizations. Software*, B Boehm , R Turner . 2005. IEEE. 22 p. .

[Beck et al. ()] *Manifesto for agile software development. The Agile Alliance*, K Beck , M Beedle , A Van Bennekum , A Cockburn , W Cunningham , M Fowler , . . Thomas , D . 2001. p. .

[Meso and Jain ()] P Meso , R Jain . *Agile software development: adaptive systems principles and best*, 2006.

[Abrahamsson et al. (2003)] 'New directions on agile methods: a comparative analysis'. P Abrahamsson , J Warsta , M T Siponen , J Ronkainen . *Proceedings. 25th International Conference on*, (25th International Conference on) 2003. May. 2003. Ieee. p. . (Software Engineering)

[Charvat ()] *Project management methodologies. Selecting, implementig, and supporting*, J Charvat . 2003.

[Paetsch et al. (2003)] 'Requirements engineering and agile software development'. F Paetsch , A Eberlein , F Maurer . *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on*, 2003. June. IEEE. p. .

[Leffingwell ()] *Scaling software agility: best practices for large enterprises*, D Leffingwell . 2007. Addison-Wesley Professional.

[Leffingwell ()] *Scaling software agility: best practices for large enterprises*, D Leffingwell . 2007. Addison-Wesley Professional.

[Schwaber and Beedle ()] K Schwaber , M Beedle . *Agile software development with Scrum*, 2002. Prentice Hall. 18. (PTR Upper Saddle River^eNJ NJ)

[Greer and Ruhe ()] 'Software release planning: an evolutionary and iterative approach'. D Greer , G Ruhe . *Information and Software Technology* 2004. 46 (4) p. .

[Highsmith et al. ()] 'The Great Methodologies Debate: Part 1'. J Highsmith , A V Traditional , M Love , N War . *The Journal* 2001. (12) p. 14.

[Highsmith et al. ()] 'The Great Methodologies Debate: Part 2'. J Highsmith , A V Traditional , M Love , N War . *The Journal* 2001. (12) p. 14.

[Coram and Bohner (2005)] 'The impact of agile methods on software project management'. M Coram , S Bohner . *ECBS'05. 12th IEEE International Conference and Workshops on the*, 2005. April. 2005. IEEE. p. . In Engineering of Computer-Based Systems

[Pikkarainen et al. ()] 'The impact of agile practices on communication in software development'. M Pikkarainen , J Haikara , O Salo , P Abrahamsson , J Still . *Empirical Software Engineering* 2008. 13 (3) p. .

[Bahli and Zeid (2005)] 'The role of knowledge creation in adopting extreme programming model: an empirical study'. B Bahli , E A Zeid . *Information and Communications Technology, 2005. Enabling Technologies for the New Knowledge Society: ITI 3rd International Conference on*, 2005. December. IEEE. p. .

[Boehm and Papaccio ()] 'Understanding and controlling software costs. Software Engineering'. B W Boehm , P N Papaccio . *IEEE Transactions on* 1988. 14 (10) p. .

[Boehm and Turner ()] 'Using risk to balance agile and plan-driven methods'. B Boehm , R Turner . *Computer* 2003. 36 (6) p. .

[Boehm and Turner ()] 'Using risk to balance agile and plan-driven methods'. B Boehm , R Turner . *Computer* 2003. 36 (6) p. .

[Dyba and Dingsoyr ()] *What do we know about agile software development? Software*, T Dyba , T Dingsoyr . 2009. IEEE. 26 p. .