My Scaled Scrum: Integrating Mega Framework and DAD

Mohamed Ahmed Radwan
Information Systems Department, Faculty of Computers and Information, Helwan University Egypt
Mohamed.radwan.msf@gmail.com
Ahmed Bahaa Farid, Associate Professor
Information Systems Department, Faculty of Computers and Information, Helwan University Egypt
Ahmed.Bahaa@compupharaohs.com
Dr. Mona Mohamed Nasr
Information Systems Department, Faculty of Computers and Information, Helwan University Egypt
drmona_nasr@yahoo.com

---ABSTRACT---

Many software organizations have moved from traditional methods for software development, such as waterfall method to usage of agile methods. Agile methods are used especially in software development and are constantly refurbishing and improving initial plans along the way. In software development the systems usually require frequent changes during the development process. This method is very suitable for small project and organizations, but it is very hard to implement it in large organizations with large projects and teams. This paper aims to identify weaknesses of two existing scrum frameworks used for large organizations and to present proposed hybrid framework scaled from both existing frameworks. It's highlighting the importance of pre-defined lifecycle of teams, which is key factor in achieving better timeline and to avoid mistakes that affects the time of release deployment.

Keywords - **Scrum, Agile Projects, Software development, Project Management**

Date of submission: July 29, 2016 Date of Acceptance: Aug 18, 2016

I. INTRODUCTION

In last year's the Agile software development approach is becoming more and more recognized and accepted. The main goal of its usage is to achieve the best quality and timeline of the project and to deploy the best possible solutions for the customer.

Our Information Society is a Complex System. We live in exciting times — times of rapid and ever-accelerating change. The changes are happening so quickly that it is difficult to keep up-to-date, much less retain control. Everything seems to be changing at once — the political map, the global economy, our institutions, human society, company structures, business practices, and individual lifestyles. And the changes all seem to be heavily inter-linked. Change triggers change [9].

According to Agile Manifesto [1], Agile is configured on a set of 12 basic principles which target as a highest priority the customer satisfaction trough early and continuous delivery of software. Other principles refer to frequent deliveries of software, close collaboration between teams and developers, self-organization, continuous improvement and small teams.

The 12 principles of Agile Manifesto are shown in figure 1:

We follow these principles:

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

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 to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals.
Give them the environment and support they need,
and trust them to get the job done.

The most efficient and effective method of
conveying information to and within a development
team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development.
The sponsors, developers, and users should be able
to maintain a constant pace indefinitely.

Continuous attention to technical excellence
and good design enhances agility.

Simplicity--the art of maximizing the amount
of work not done--is essential.

The best architectures, requirements, and designs
emerge from self-organizing teams.

At regular intervals, the team reflects on how

to become more effective, then tunes and adjusts

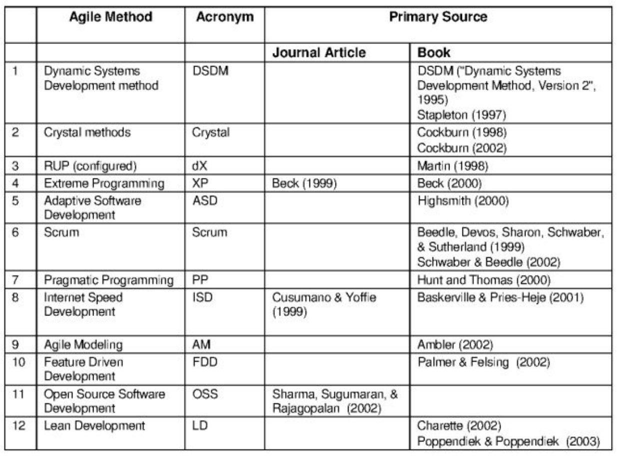
its behaviour accordingly.

Figure 1. 12 main principles in the Agile Manifesto

Agile framework is based on constant software development, which highlights the software deliveries after iteration, which shouldn't take more than two weeks [2]. Agile development strives to have fully tested software before the end of iteration and it can be understood as a way of thinking about the development itself or as a philosophy.

The essential agile development process models are Extreme Programming (XP), Scrum, Open Unified Process (OpenUP), Feature Driven Development (FDD), Agile Modeling (AM), Rational Unified Process (RUP), Dynamic Systems Development Method (DSDM) and Open Source Software development (OSSD) [2]. In comparison with traditional methods all Agile methods are providing many additional improvements as Agile methods are dividing tasks into smaller pieces with less planning and with more efficient communication and active customer's involvement in the development process [3].

In the years 1997 to 2003 twelve agile methods were published and next table is showing all those methods and some references [8].



	Agile Method	Acronym	Primary Source	
			Journal Article	Book
1	Dynamic Systems Development method	DSDM		DSDM ("Dynamic Systems Development Method, Version 2", 1995) Stapleton (1997)
2	Crystal methods	Crystal		Cockburn (1998) Cockburn (2002)
3	RUP (configured)	dX		Martin (1998)
4	Extreme Programming	XP	Beck (1999)	Beck (2000)
5	Adaptive Software Development	ASD		Highsmith (2000)
6	Scrum	Scrum		Beedle, Devos, Sharon, Schwaber, & Sutherland (1999) Schwaber & Beedle (2002)
7	Pragmatic Programming	PP		Hunt and Thomas (2000)
8	Internet Speed Development	ISD	Cusumano & Yoffie (1999)	Baskerville & Pries-Heje (2001)
9	Agile Modeling	AM		Ambler (2002)
10	Feature Driven Development	FDD		Palmer & Felsing (2002)
11	Open Source Software Development	OSS	Sharma, Sugumaran, & Rajagopalan (2002)	
12	Lean Development	LD		Charette (2002) Poppendiek & Poppendiek (2003)

Figure 2. Agile methods published from 1997 to 2003

But not all methods contributed to Agile Manifesto and of course not all equally. The next figure is representing all Agile Manifesto contributed methods [8].

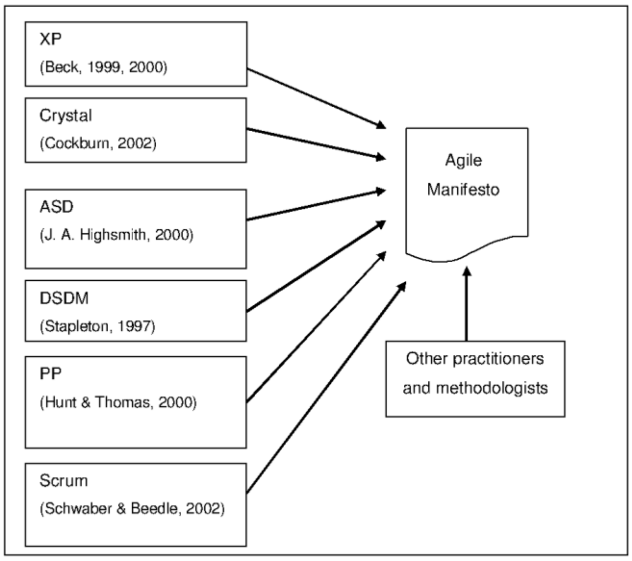


Figure 3. Contributed methods to the Agile Manifesto

II. RELATED WORK

This chapter of the paper will focus on research and analyses or study cases related too Agile and it will mainly focus two specific scaled frameworks: Mega Framework and Disciplined Agile Delivery (DAD) framework.

A. *Agile*

The Agile methodology was contained in a form of "manifesto" by group of 17 process methodologists, who held a meeting in order to achieve better way for software development [1]. The manifesto is published at official website and highlights the next values [1]:

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

Even by the word itself it can be understood that it is meaning flexibility and responsiveness, so agile methods signify the ability to survive in atmosphere of continuous change and development with success [6]. According to Cockburn this feature of maneuverability is very important it today's software development, where the competition is fast and more hard than ever before, because of continuous development and market demands [7].

Agile software development is considered to be an extensive standard in software engineering and it has been accepted by the industry in great scope. It uses the customary technique for continuous improvements, with software methodologies following the generic engineering standard of requirements, design, build, and maintain. The main focus in on rapid development, considerable amount of simulations releases of the software with high-quality produce. Agile methods are usually used in software development to help businesses respond to vulnerability efficiently [9].

In Agile Software Development are five categories of factors, which affects success: Organizational, People, Process, Project, and Technical [10]. Each of those factors is divided into subdivision of other factors.

B. *Mega Framework*

By case study provided from students of Universo Online São Paulo, Brazil, the Mega Framework is known as a set of many practices and meetings, which are providing synchronization in all levels [4]. By this case study the Mega Framework's strategy refers to:

- Feature teams
- Mega Backlog
- Grow, then split
- Hiring and ramp up
- Sprint length
- Teams per release
- Values instead of rules
- Development environment
- Continuous improvement

The main framework of Mega Frameworks by this study represents:

- Mega Planning
- Mega Stand-up
- Mega Retrospective
- Sprint Reviews
- Weekly Pre-Planning
- Weekly Product Owner and ScrumMaster meeting
- Regular Mega meetings with business area
- Knowledge sharing

The key factors of this scaled framework is to have one main, huge backlog for all teams. Each team should have assigned team leader, who should be responsible with communication with ScrumMaster. In the middle of the Sprint, the teams will have a "Mega StandUp" meeting in order to sync all teams [4].

C. *DAD Framework*

The definition of Disciplined Agile Delivery, DAD framework is stating that this is an Agile framework, which is people and learning oriented, in order to provide IT delivery with a target oriented scalable lifecycle [5]. By this definition the DAD is a hybrid framework which uses different practices from already existing methods, such as Scrum, Extreme Programming (XP), Kanban, and Agile Modeling (AM) and provides suggestions how and in which cases is the best option to apply them together.

III. WEAKNESSES IN MEGA AND DAD FRAMEWORK

The study of DAD framework is showing concerns about handling simultaneously different lifecycles. The main concern here is that the team members needs to be flexible enough to adopt the delivery lifecycle. The key factor is having the right resources, which are flexible to adopt to the team's lifecycle.

With the Mega framework the main concern is connected to the timeline and mistakes in process of release deployment. Another concern is simultaneous management of different releases. As both concerns could lead to late and slow delivery, this paper is striving to solve both issues with tailored approach in proposed framework.

IV. PROPOSED FRAMEWORK

The proposed model is representing an extended version of Mega framework, which is using some of the key features of DAD framework. The main idea for this scaling is that Mega framework is designed for large organizations, with different teams and one backlog, with key factor in communication between all areas.

Existing framework doesn't provide any clear values to define the lifecycle of different teams, which should be defined before the actual planning in order to achieve better timeline and to avoid mistakes that affects the release time deployment.

The new lifecycle, which should be done in very first phase, before the actual planning would allow teams to understand what is actually needed. The education phase is actually a meeting on a highest level, between developer management, clients and even some other client's management, in order to achieve the better understanding of the project. This is necessary to create the best output of the software development and to set the goals, which needs to be met in the later phases of the project development. The customer is highly involved with the development team in order to classify the expected features of the system, which are valuable to the customer. The development team or management needs to give an estimation about time duration and to identify if the features are achievable.

In the phase of initial idea, the team should first very quickly explore what is needed from educational phase, via series of quick experiments, which will show either to fail fast of to continue with deployment of initial idea. First the teams will start with very small build, then they will try to deploy it and after the review, the result will show if the team should continue with development of initial idea or should they abort the development and deployment.

The proposed model is shown in figure 2:

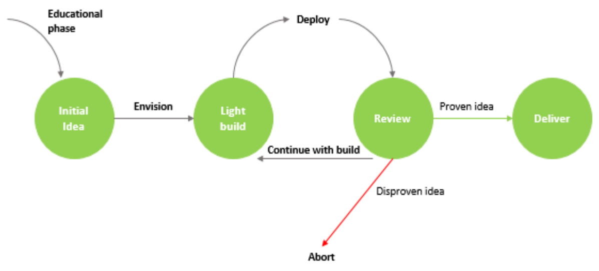


Figure 4. Proposed model framework

The process of proposing initial idea is done before initial planning for the team's lifecycle. This is taking in consideration that first tasks are designed for development of idea and defining the priorities in basic level of development. Even further, it is classified as a development of comprehensive model process which is assigned on development level. This phase requires approaching interaction between Product Owner, Scrum master and scrum team. The timing of this phase is equal to a single iteration or sprint length to have a light picture about all activities for legitimate time frame and lifecycle of the project.

V. FUTURE WORK AND CONCLUSION

This paper pursuit the review of scaling agile methodologies in large organizations with software development projects. It presents two frameworks with all their dimensions to reach the best outputs in software development. Due to efficient delivery, customer satisfaction factors and quality developed software, agile methodologies are very prominent in software development. The focus of agile is on adopting the practices during the development process and to be flexible to change the idea whenever it is required to achieve customer requirements and organizational objectives. Furthermore, the paper represents the proposed scaling of Mega framework in order to meet better timeline of single and final release. The flow chart represents the performance of initial step incorporated into existing framework. As the proposed framework is representing only a theoretical way, there needs to be done also practical implementation on different agile software development cases. The future work is to test the proposed model in appropriate software environments and to perform statistical studies, which will show the additional risks and opportunities.

By that means the main objectives were founded to validate this proposed framework. These objectives are as follows.

Objective 1: Optimized communication between teams

Objective 2: Enabling normal system workflow in the new framework

Objective 3: Lower the reliance between the different parts

Objective 4: Avoid the dependencies between different parts of framework

For each objective a set of essential questions were developed to define it. The questioner for each objective contained 30 questions, designed by IT specialists and developers. All results for all objectives used the same gathering, analyzing and scaling process. The scaling process that was used is known as Likert scaling system (Figure 5), which is using usually five-level Likert item and it's is a bipolar scaling method, measuring either positive or negative response to a statement [11].

Very low
Low
Normal
High
Very high

Figure 5. Likert scale

The output of the survey, the detailed responses of participants to each objective is described in details in next sections. All results are showing the cumulative response in %.

Objective 1: Optimized communication between teams

The table shown in Figure 6 is showing all cumulative responses for objective 1. We can see that 23,8% of all participants responded as a very high and 26,8% responded as a high, which shows a good result. Only 8,5 % of participants responded as a very low level of communication.

Question nr.	Very low	Low	Normal	High	Very high
#1	10	22	45	23	0
#2	5	5	40	35.5	14.5
#3	20	14	15.5	10	40.5
#4	0	0	30	40	30
#5	7.5	18	15	25.5	34
Total	**42.5**	**59**	**145.5**	**134**	**119**
Average	**8.5**	**11.8**	**29.1**	**26.8**	**23.8**

Figure 6. Cumulative response for objective 1

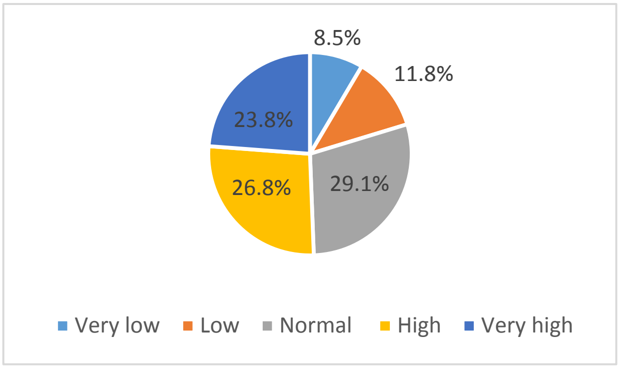


Figure 7. Average cumulative response for objective 1

Objective 2: Enabling normal system workflow in the new framework

The table shown in Figure 8 is displaying all cumulative responses for objective 2. We can see that 34,9% of all participants responded as a very high and 23,7% responded as a high, which is representing a higher proportion. Only 6,4 % of participants responded as a very low level of system workflow and 12,8% as a low.

Question nr.	Very low	Low	Normal	High	Very high
#6	0	22	45	23	10
#7	2	10	15.5	20	52.5
#8	20	14	15.5	10	40.5
#9	10	0	20	40	30
#10	0	18	15	25.5	41.5
Total	**32**	**64**	**111**	**118.5**	**174.5**
Average	**6.4**	**12.8**	**22.2**	**23.7**	**34.9**

Figure 8. Cumulative response for objective 2

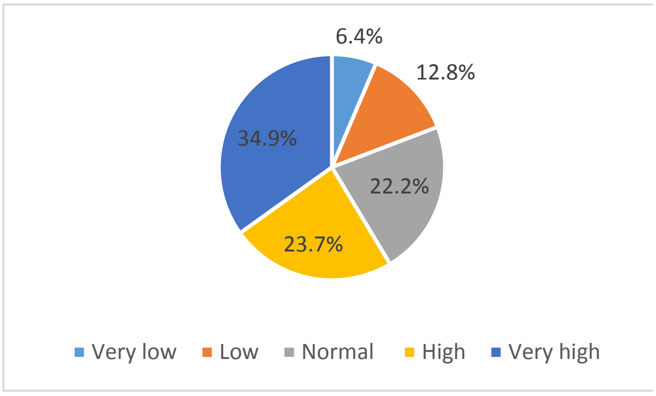


Figure 9. Average cumulative response for objective 2

Objective 3: Lower the reliance between the different parts

The table shown in Figure 10 is displaying all cumulative responses for objective 3. It is showing that only 15,8 % of all participants responded as a very high and in the biggest scope, they responded as normal with 26%.

Question nr.	Very low	Low	Normal	High	Very high
#11	20	13	20	44	3
#12	3	5	20	15	57
#13	15.5	33	25.5	9	17
#14	30	15	33.5	21.5	0
#15	19	44	31	4	2
Total	**87.5**	**110**	**130**	**93.5**	**79**
Average	**17.5**	**22**	**26**	**18.7**	**15.8**

Figure 10. Cumulative response for objective 3

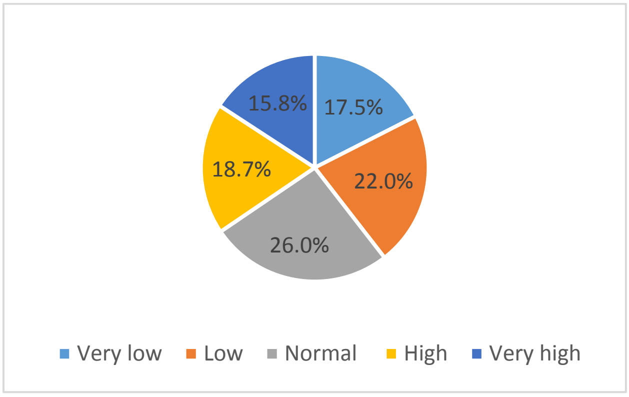


Figure 11. Average cumulative response for objective 3

Objective 4: Avoid the dependencies between different parts of framework

The table shown in Figure 12 is displaying all cumulative responses for objective 4. It is showing that only 26,3 % of all participants responded as a very high and in the biggest scope, they responded as high with 26%. The proportion of participants with low or very low repose is just a little bit over 20%.

Question nr.	Very low	Low	Normal	High	Very high
#16	10	22	45	23	0
#17	0	13	22.5	21	43.5
#18	6.5	27	8	40	18.5
#19	9	0	14	47	30
#120	3	18.5	22	17	39.5
Total	**28.5**	**80.5**	**111.5**	**148**	**131.5**
Average	**5.7**	**16.1**	**22.3**	**29.6**	**26.3**

Figure 12. Cumulative response for objective 4

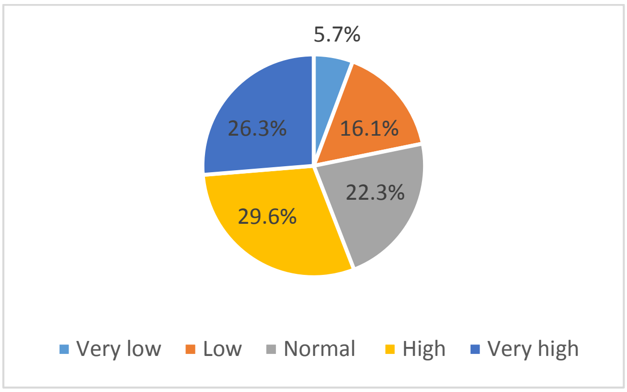


Figure 13. Average cumulative response for objective 4

Overall results in all settled objectives are showing that the biggest proportion are almost equally divided to very high, high and normal with the proposed framework.

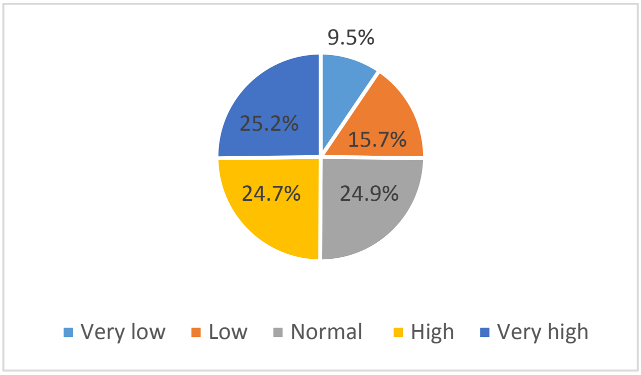


Figure 14. Overall cumulative responses for all objectives

ACKNOWLEDGEMENTS

An acknowledgement section may be presented after the conclusion, if desired.

REFERENCES

[1] Agile Manifesto (2011). The Agile Manifesto, online, available on: http://agilemanifesto.org/. [Accessed: 10th August 2016].

[2] Pressman R.S. Software Engineering. McGraw Hill, USA, 2010

[3] Chow, T., Cao, D., 1984, A survey study on critical success factors of agile software, Journal of Systems and Software, Volume 81, Issue 6, Pages 961-971.

[4] Scaling Scrum Step by Step: "The Mega Framework", online, available on: https://agilealliance.org/wp-content/uploads/2016/01/scaling-scrum-mega-framework.pdf [Accessed: 10th August 2016]

[5] Disciplined Agile Delivery, online, available on: http://www.disciplinedagiledelivery.com/introduction-to-dad/. [Accessed: 10th August 2016]

[6] Anderson, D.J., 2004. Agile Management for Software Engineering. Prentice Hall, Upper Saddle River, New Jersey.

[7] Cockburn, A., 2002. Agile Software Development. Addison-Wesley, Boston, Massachusetts.

[8] Diane Elizabeth Strode, 2005, Thesis of Master of information Science, Massey University, Palmerstone North, New Zealand

[9] Ioannis Antoniou., (1999), "The Information Society as a Complex System", Journal of Universal Computer Science, vol. 6, no. 3 (2000), 272-288, No.1, pp.1-pp. 6.

[10] Magazine. <http://www.sdmagazine.com/documents/s=844/sdm0108a/0108a.htm> (accessed December 2005).

[11] Likert scale https://en.wikipedia.org/wiki/Likert_scale (accessed on 12 August 2016)