International Journal of Engineering Research & Management Technology
ISSN: 2348-4039
January- February 2023 Volume-10, Issue-1
Email: editor@ijermt.org www.ijermt.org

Evaluation of Processes of Lightweight Methodologies: Developers' Perspective

Varsha Sud[1], Arvind Kalia[2], Hardeep Singh[3], Naveen Kumar[4]

[1 2 4] Department of Computer Science, Himachal Pradesh University, Shimla

[3] Department of Computer Science, Guru Nanak Dev University, Amritsar

varshasud14@gmail.com, arvkalia@gmail.com, hardeep.dcse@gndu.ac.in, nkjaglan@gmail.com

ABSTRACT

In this fast growing era of software industry, it is the need of the hour to designed and develop the software quickly and at a fast pace. To design such software the new methodologies were devised which are popularized as agile or lightweight methodologies. Various agile models came into existence. In this paper, the phases or processes of three of these models (XP, SCRUM, FDD) were evaluated on the parameters: (i) user involvement, (ii) time devoted to phases and (iii) training provided to the users. For evaluating these models a questionnaire for lightweight methodologies was designed. The responses of the software developers were taken and analyzed by using various statistical tools.

KEYWORDS: Lightweight, XP, SCRUM, FDD.

I. INTRODUCTION

From its beginnings in 1940s, software development has evolved greatly. In the early age of computer, its use was limited to fast and accurate processing of data related to few areas like defense, scientific applications etc. As the impact of software started growing on society, the need to develop efficient and accurate software became obvious. The new technologies were developed with spectra of objectives to make software easy to understand, design, develop and validate. The fast and cost effective software development is also high in the priority of the objectives. In the early days, most of the designed software used to fail due to the non-availability of the set techniques and procedures to develop the software. Software engineering was evolved as a separate field to deal with the systematic development of software. Off late software systems are becoming more complicated than ever before. One of the major handicaps of the traditional SDLC model is that until the first phase is complete the application does not proceed to the next phase and if by chance there are some changes in the later stage of the cycle it becomes very challenging to implement those changes as it would involve revisiting the earlier phase and carrying out the changes. Thus for handling modern age, large and complex software systems and keeping pace with the continuous changing customer needs and requirements agile methods were developed (Shore, 2007).

Agile breaks down larger project into small manageable chunks called iterations. At the end of each phase, the product so obtained is shown to the users or stakeholders for their feedback before entering the next phase of software development. It uses incremental, iterative work sequences that are commonly known as sprints. Agile methodology promotes continuous iteration of development /testing throughout the software development where both development and testing activities run concurrently. The lightweight strategies allow the developers to build the software more effectively and efficiently. The lightweight strategies are more responsive to the changes that are happening in the business. These strategies mainly emphasis on short life cycles, these are simple and development oriented. These models focused more on the participation of the team that is developing the software. Various lightweight software development process models are: Extreme Programming (XP), SCRUM, Feature Driven Development (FDD), Dynamic System Development Method (DSDM), Adaptive Software Development (ASD) and CRYSTAL (Daba et al., 2008).

II. LITERATURE REVIEW

Many studies have been conducted on the agile and lightweight methodologies by various researchers. Some of the important studies are referred as under:

Sarker et al. (2015) explained that software being a significant part of modern society various software development process models were evaluated. Software process model is a description of the sequence of activities carried out in a software engineering project and the relative order of these activities. The main objective of their study was to represent different models of software development and different aspects of each model to help the developers to select specific model at specific situation depending on customer demand (Sarker et al., 2015).

Sharma et al. (2012) focussed on the comparative study of agile processes. It served as guide to other software developers about various process models. Agile processes have important applications in the areas of software project management, software schedule management, etc. In particular the aim of agile processes is to satisfy the customer, faster development times with lower defects rate. The comparison of the agile processes with other software development life cycle models was presented. Agile processes are not always advantageous, they have some drawbacks as well. The advantages and disadvantages of agile processes were also discussed (Sharma S. et al., 2015).

Avasthy (2017) discussed about the breakthrough caused by agile on conventional development methodology. Their study discussed about the breakthrough caused by Agile in the Conventional development methodology. It will give an idea about Agile Mindset and Enterprise Transformation needed to adopt Agile. It deeply focused on Agile variant Scrum and its implementation. It analyzed different Agile Metrics and examined their impact on Development Lifecycle of a product. It is concluded by explaining the Industrial shift towards Agile and gives a sneak-peak into the future of Agile (Avasthy, 2017).

Nawaz et al. (2021) made a comparative survey between traditional approaches and agile techniques used for software development. A comparison of agile methodologies was also performed in detail to highlight their various aspects. Their article may facilitate the researchers to decide on which method is most suited to their problem. The analysis of different agile techniques was also performed to understand the positive and negative points of various Agile methods like Scrum, XP, Kanban, Feature Driven Development and Dynamic System Development System and select the most appropriate technique suited to their projects (Nawaz et al., 2021).

Nagpal (2019) compared various Agile Methodologies like Extreme Programming, Scrum, Feature driven Development, Dynamic Systems Development Method and Crystal. on the basis of the nature of the project, development team skills, customer involvement, iteration duration, as well as project constraints. Their research provided an insight into the various Agile methodologies. It also highlighted the factors for the popularity of Agile approach like Reduced risks, higher customer satisfaction, in creased project control, high quality product, early and predictable delivery, focus on business values, predictable cost etc (Nagpal, 2019).

III. RESEARCH METHODOLOGY

To meet out the objective of the study a questionnaire was designed for Lightweight Methodologies. The questionnaire was administered personally and through Google forms. The software developers of various organizations irrespective of their experiences participated in the study. The analysis of data was done using SPSS and R language.

IV. ANALYSIS

In the lightweight methodology the involvement of the user and the team members is very high and very less documentation is required. The lightweight strategies are more responsive to the changes that are happening in the business. These strategies mainly emphasis on short life cycles, they are simple and development oriented. The three lightweight software development process models which are commonly used by the developers have been considered for evaluation out of various existing agile models. These three models are: Extreme Programming (XP) Model, SCRUM Model and Feature Driven Development (FDD) Model. To evaluate the process of the lightweight methodologies different three parameters have been selected for the study i.e. (i) *Involvement of the Users* (ii) *Time devoted to each development phase and (iii) Training provided to the users.*

i) Involvement of the Users: The developers were asked that how frequently they involved the users during the different phases of system development while using XP, SCRUM and FDD methodologies. The developers responses were taken on five point scale. The average scores as per the responses of the developers were computed and are presented in Table 1, Table 2 and Table 3, respectively.

Table 1: Involvement of Developers in System Development Phases of XP
(Values in Percentage)

Phases / Involvement	Always	Mostly	Average	Some of the times	Never
Exploration	32	28	19	13	8
Planning	24	35	25	11	5
Iterations to Release	13	27	41	14	5
Productionizing	21	28	25	18	8
Maintenance	17	32	28	12	11
Death	8	14	43	27	8
Any Other	19	19	39	14	9

It is evident from the Table 1 that most of the developers, while developing the software with XP, involved the users, for whom they were developing the software, in most of the phases of XP model. However, in the 'Death' phase, the user involvement is very less and below average.

It can be observed from the Table 2, that the developers developed software using SCRUM, involved the users very frequently during 'Product Backlog', 'Sprints', 'Sprints Planning Meeting' & 'Daily Scrum', whereas, the user involvement is less during the 'Sprints Backlog'.

Table 2: Involvement of Developers in System Development Phases of SCRUM
(Figures in Percentage)

Phases / Involvement	Always	Mostly	Average	Some of the times	Never
Product Backlog	27	27	28	15	2
Sprints	24	34	25	14	4
Sprints Planning Meeting	21	24	36	13	6
Sprints Backlog	11	24	29	28	7
Daily Scrum	20	21	23	12	17
Any Other	19	19	29	19	14

It can be noted from Table 3 that the users were involved very frequently by the developers on FDD methodology during 'Develop an Overall Model', 'Build a Feature List' and 'Design by Feature' phases. During 'Plan by Feature' phase the users involvement was a bit reduced but it was still above average. During 'Build by Feature' phase, the involvement of the user was further reduced by the developers and it can be said that this involvement was below average.

Table 3: Involvement of Developers in System Development Phases of FDD
(Figures in Percentage)

Phases / Involvement	Always	Mostly	Average	Some of the times	Never
Develop an Overall Model	22	29	36	10	3
Build a Features List	16	30	38	14	3
Plan by Feature	9	30	34	19	8
Design by Feature	21	24	27	17	11
Build by Feature	11	21	28	25	14
Any Other	12	4	44	24	16

ii) Time devoted to each Development Phase:

The time devoted by the developers on each of the phases of XP, SCRUM and FDD models for software development are presented in TABLE 4, Table 5 and Table 6, respectively. A majority of the developers using XP devoted 11-20% of the total time for 'Exploration' phase, 21-30% of the time for 'Planning', 'Iterations to Release', 'Maintenance' & 'Death' phases, whereas 31-40% of the time to 'Productionizing' phase. On the whole, when the total time devoted on various phases of XP are compared then it can be said that 'Exploration' phase needed less time followed by 'Planning' and 'Death' phases, whereas, most of the time of software development was devoted to 'Productionizing' and 'Maintenance' phase.

Table 4: Time devoted to each Phase in Software Development of XP
(Figures in Percentage)

Phases / Time	0-10 %	11-20 %	21-30 %	31-40 %	41-50 %
Exploration Phase	19	37	23	15	6
Planning Phase	14	21	32	24	9
Iterations to Release Phase	1	28	36	27	8
Productionizing Phase	6	6	29	38	21
Maintenance Phase	3	20	35	22	20
Death Phase	22	22	36	13	7
Any Other	15	15	40	15	15

For SCRUM, almost equal number of developers were of the view that 11-20% & 21-30% of total time needs to be devoted to ’Product Backlog’. A few of these developers devoted less than ten percent and 31-40% of the time. For 'Sprint', ’Sprints Planning Meeting' and 'Daily Scrum' majority of the developers devoted 21-30% of the time, whereas almost equal number of developer devoted 11-20%, & 31-40% of the time for 'Sprints Backlog'. On 'Sprints Planning Meeting' most of the developers devoted 21-30% of the time, whereas a good number of developers also devoted for 11-20% & 31-40%. A few of the developer spent less than ten percent of system development time on ’Sprints Planning Meeting’. So, 'Sprints', 'Sprint Planning Meeting' and 'Daily Scrum' were time consuming phases of SCRUM.

Table 5: Time devoted to each Phase in Software Development of SCRUM
(Figures in Percentage)

Phases / Time	0-10 %	11-20 %	21-30 %	31-40 %	41-50 %
Product Backlog	19	31	33	14	3
Sprints	2	25	33	22	18
Sprints Planning Meeting	12	21	42	18	7
Sprints Backlog	13	28	28	18	13
Daily Scrum	13	22	31	15	19
Any Other	17	13	39	22	9

In the opinion of most of the developers using FDD, it can said that upto 30% of the time devoted to each of the phases 'Develop an Overall Model', 'Build a Features List', and 'Design by Feature', whereas 'Plan by Feature' and 'Build by Feature' may require more time to complete.

Table 6: Time devoted to each Phase in Software Development of FDD

(Figures in Percentage)

Phases / Time	0-10 %	11-20 %	21-30 %	31-40 %	41-50 %
Develop an Overall Model	15	25	31	19	11
Build a Features List	9	31	32	22	5
Plan by Feature	4	24	38	28	6
Design by Feature	7	30	36	26	11
Build by Feature	9	10	34	29	18
Any Other	13	7	47	13	20

iii) *Training provided to the users:*

After the development of the software, users must be provided with the training on the developed software so that they can use it properly. The responses of the developers about the duration of the training provided to the users on the developed software using XP, SCRUM and FDD are tabulated in TABLE 7, Table 8 and Table 9, respectively. A large number of XP developers (45%) provided a training of '1-3 Weeks' to their users, whereas, 34% of the developers provided '4-8 Weeks' of training. Only a few of them responded for '9-12 Weeks' of training to the users.

Table 7: Required Training using XP

(Figures in Percentage)

Duration	No Training	1-3 Weeks	4-8 Weeks	9-12 Weeks	More than 12 Weeks
Developers	4	45	34	16	1

The majority of the SCRUM developers (47%) provided '1-3 Weeks' of the training to the users on the developed software whereas a good number these of developers (35%) provided '4-8 Weeks' of training to their users. There are very few developers (7%) who think that no training was required for the users on the software development by them using SCRUM methodology may be due to the reason that the user were involved at each phase of the development.

Table 8: Required Training using SCRUM

(Figures in Percentage)

Duration	No Training	1-3 Weeks	4-8 Weeks	9-12 Weeks	More than 12 Weeks
Developers	7	47	35	9	3

Majority of the developers using FDD them (39%) provided '1-3 Weeks' training, whereas almost equal number of developers (37%) provided '4-8 Weeks' of the training. A few of them (19%) also responded that '9-12 Weeks' training was required for the users.

Table 9: Required Training for FDD

(Figures in Percentage)

Duration	No Training	1-3 Weeks	4-8 Weeks	9-12 Weeks	More than 12 Weeks
Developers	5	39	37	19	0

CONCLUSION

On the basis of the discussion, it may be concluded that most of the developers of all the models involved the users in all the phases, except for 'Death' phase in XP, 'Sprint Backlog' in SCRUM and 'Build by Feature' in FDD. Further it can be inferred that most of the system development time was spent on 'Productionizing' followed by 'Maintenance' phase of Extreme Programming and minimum time is spent on 'Exploration' phase. Similarly, most of the system development time was spent on 'Sprint', 'Sprint Planning Meeting' and 'Daily Scrum' by the developers and the least time on "Sprints Backlog' using SCRUM and most of the system development time was spent on 'Build by Feature' phase of FDD. The developers provided '1-8 Weeks" of training to the users irrespective of the methodology used by them.

REFERENCES:

Aggarwal, K. K., & Singh, Y. (2005). *Software Engineering*. New Age International, New Delhi.

Avasthy, A. (2017). Systematic study on Agile Software Metrics. *International Journal of Computer Science and Information Technologies*, *8*(5), 552–555.

Dyba, T., & Dingsoyr, T. (2008). *Empirical studies of agile software development: A systematic review* (First, Vol. 50). Information and Software Technology (Elesvier Inc.).

Nagpal, P. (2019). A Path for Assessing Better Agile Methodology. *International Journal of Research in Advent Technology*, *7*(5), 422–425.

Nawaz, M., Nazir, T., Islam, S., Masood, M., Mehmood, A., & Kanwal, S. (2021). Agile Software Development Techniques: A Survey. *Proceedings of the Pakistan Academy of Sciences: Part A, 58(1)*, 17–33.

Sarker, I. H., Faruque, F., Hossen, U., & Rahman, A. (2015). A Survey of Software Development Process Models in Software Engineering. *International Journal of Software Engineering and Its Applications*, *9*(11), 55–70. http://dx.doi.org/10.14257/ijseia.2015.9.11.05

Sharma, S., Sarkar, D., & Gupta, D. (2012). Agile Processes and Methodologies: A Conceptual Study. *International Journal on Computer Science & Engineering*, *4*(5), 892–898.

Shore, J., & Warden, S. (2007). *The Art of Agile Development* (First). O'Reilly.

Singh, R., Kumar, D., & Sagar, B. B. (2017). Interpretive Structural Modelling in Assessment of Agile Methodology. *IEEE*, 1–4.

Software Metrics | Types of Software Metrics with Diagram. (2020, April 3). *EDUCBA*. https://www.educba.com/software-metrics/

Sunner, D. (2016). Agile: Adapting to need of the hour: Understanding Agile methodology and Agile techniques. *2nd International Conference on Applied and Theoretical Computing and Communication Technology (ICATccT)*, 130–135.

Szalvay, V. (2004). An Introduction to Agile Software Development. *Danube Technologies*.

http://www.danube.com.

Tanveer, B., Guzman, L., & Engel, U. M. (2017). Effort estimation in agile software development: Case study and improvement framework. *John Wiley & Sons, Inc.*, 1–14. https://doi.org/10.1002/smr.1862

Thulasee, K. S., Sreekanth, S., Perumal, L., Reddy, K., & Kumar, R. (2012). Explore 10 Different Types of Software Development Process Models. *International Journal of Computer Science and Information Technologies*, *3*(4), 4580–4584.

Tolfo, C., Wazlawick, R. S., Ferreira, M. G. G., & Forcellini, F. A. (2018). Agile practices and the promotion of entrepreneurial skills in software development. *John Wiley & Sons, Inc.*, 1–23. https://doi.org/10.1002/smr.1945