International Journal of Advanced Research in Engineering Technology & Science ISSN: 2349-2819
Email: editor@ijarets.org Volume-9, Issue-6 June – 2022 www.ijarets.org

# Heavyweight and Lightweight Models: An Analysis

**Varsha Sud[1], Arvind Kalia[2], Hardeep Singh[3]**

[1 2] 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

**ABSTRACT:** In this era digitization everything revolves around the software. With the numerous and varied application areas like transportation, security, Artificial Intelligence, Internet of Things, banking, the software development has become complex, tedious, time consuming task. To meet out such complexities various new software development methodologies emerged from time to time. The developers stared from heavyweight methodologies and are now proceeding towards lightweight methodologies. In this paper, a comparison of various heavyweight and lightweight methodologies has been presented. For making the comparison, two different questionnaire, one each for heavyweight and lightweight methodologies have been designed. The responses, of the software developers were taken and analyzed by using various statistical tools.

***KEYWORDS: SDLC, Heavyweight, Lightweight, XP.***

**I. INTRODUCTION:** In this era of digitization everyone is surrounded with software heavily, be it domestic appliances, communication systems, security systems, business applications, AI, satellite launching etc. Developing software is a challenging task because of its complex nature. In Software Engineering terminology according to IEEE standard Glossary, the software life cycle is "The period of time that starts when a software product is conceived and ends when the product is no longer available for use". A software process also known as software methodology is a set of related activities that lead to the production of the software.

Software Engineering has brought revolution in every arena of software development. With the increase in dependency on software in every aspect of life, its correctness and completeness were also highly required and desired. Software development is guided by different models of software engineering. New software development models and the techniques have emerged and made the analysis, design, testing and implementation processes even more challenging.

Traditional software development methodologies, such as the Waterfall model, were prevalent at the time but often faced challenges in delivering software projects efficiently and adapting to changing requirements. Agile emerged as a response to these limitations and aimed to create a more flexible and customer-centric approach to software development.

The heavyweight software development strategies are also termed as traditional Software Development Life Cycle (SDLC). This method has a systematic approach towards software development. The traditional software development traverses through some specific sequence. The cycle starts with problem identification, feasibility analysis, requirement specifications, system analysis, building software, testing, deploying and maintenance. This type of methodology mainly deals with heavy documentation, detailed planning and design. Various Heavyweight Software Development Process Models are: Waterfall, Incremental, Prototype, Rapid Application Development (RAD), Spiral and V-Shaped.

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.

**II. LITERATURE REVIEW:** Many studies have been conducted on heavyweight and lightweight methodologies by various researchers. Some of the important studies are referred as under:

Pawar at el. (2015) focused on traditional SDLC method like Waterfall model and a brief comparative study of same with Agile. Traditional SDLC model is sequential development methodology where output of one development phase becomes input for other. Agile software development methodology is widely used software development process which overcomes the drawbacks of traditional software development methods. The primary data collection method was interviews of the industry expertise. The secondary source of data is reference books and Internet articles (Pawar et at., 2015).

Leau et al. (2012) emphasized the importance of Software development life cycle in software development considering it as one of the most important element in software development. SDLC depicts the necessary phases in software development. A review of traditional methods and agile methods was carried explaining the advantages and disadvantages of both methodologies. The improvements for current agile development so were suggested that this lightweight SDLC could be adopted (Leaut et at., 2012).

Popli et al. (2013) stated that use of Agile model has increased for software development. Companies are drifting from traditional Software Development Life Cycle models to Agile environment for the purpose of attaining quality and for the sake of saving cost and time. A common life cycle approach was proposed for different kinds of teams and described a mapping function for mapping of traditional methods to Agile method (Popli et al., 2013).

Rani and Bajwa (2017) were of the view that, many software development companies were using agile methodologies like scrum, extreme programming, lean software development etc. rather than traditional software development approaches. These agile practitioners claim that they had less amount of failure of projects as well as software product quality has been immensely improved as compared to traditional development. Heavyweight methodologies had many quality assurance techniques but these techniques were more oriented towards heavy reporting and these techniques had a large number of inspection methods where as agile methodologies are known for their built in quality assurance activities and management system. Many organizations are shifting towards agile software development techniques as compare to heavyweight practices. But in this scenario, still there are issues related with the quality (Rani and Bajwa, 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 detailed to highlight their various aspects. The analysis of different agile techniques was performed directly helping the researchers to understand the positive and negative points of various Agile methods and select the most appropriate technique suited to their projects (Nawaz et al., 2021).

**III OBJECTIVE:** The major objective of the study was to compare the heavyweight and lightweight software process models so that it may be easy for the developers to decide on the selection of the models for developing a successful software.

**IV. RESEARCH METHODOLOGY:** To meet out the objective of the study two different sets of questionnaires were designed, one each for Heavyweight and Lightweight Methodologies. The questionnaires were administered personally and through Google forms. The software developers of various organizations irrespective of their experiences participated in the study. The data was analyzed using SPSS and R language.

**V. ANALYSIS:** For the purpose of study, six most commonly used models each for heavyweight and lightweight methodologies were considered. Heavyweight methodology included: Waterfall Model, Incremental Model, Prototype Model, Rapid Application Development (RAD) Model, Spiral Model and V-Shaped Model and for lightweight methodology: Extreme Programming (XP) Model, SCRUM Model, Feature Driven Development (FDD) Model, Dynamic System Development Method, ASD and CRYSAL. To compare the heavyweight and lightweight methodologies four parameters have been selected for the study i.e. (i) *Use of Software Development Model*, (ii) *Use of Methodologies for various Applications, (iii) Efficiency of Methodologies for developing the software, and (iv) Impact of Models on Software Quality.*

*i) Use of Software Development Models:* The developers used to develop the software using various heavyweight and lightweight models. The choices of the developers were presented in TABLE 1 and FIGURE 1.

Table 1: Models used by Developers for Software Development
(Figures in Percentage)

| **Heavyweight Methodologies** | | | **Lightweight Methodologies** | |
|---|---|---|---|---|
| **Models** | **Developers** | | **Models** | **Developers** |
| Waterfall | 41 | | XP | 15 |
| Incremental | 22 | | SCRUM | 33 |
| Prototype | 13 | | FDD | 17 |
| RAD | 9 | | DSDM | 12 |
| Spiral | 11 | | ASD | 16 |
| V-Shaped | 4 | | CRYSTAL | 7 |

Most of the developers were using ’Waterfall’ model followed by ’Incremental’ model during the development of software using Heavyweight methodology. A few of them used ’Prototype’, and ’Spiral’ models. ’RAD’ and ’V-Shaped’ models were used only by 9% and 4% developers. In Lightweight methodology SCRUM model is the first preference of the developers to develop the software, 33% of the developers used 'SCRUM' followed by 'FDD' (17%), 'ASD' (16%), 'XP' (15%), 'DSDM' (12%) and 'CRYSTAL' (only 7%) models to develop the system.

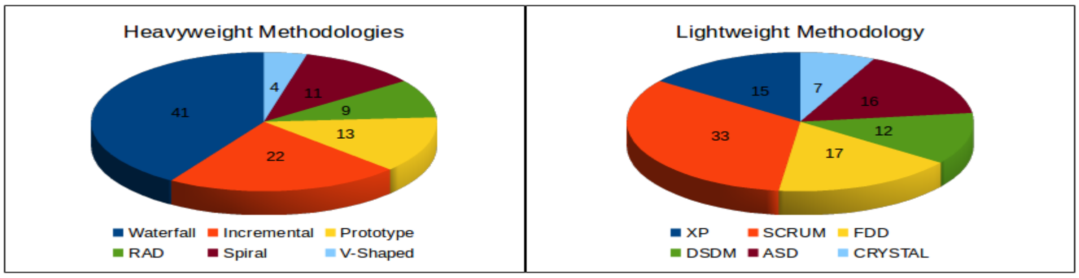

Figure 1: Models used by Developers for Software Development

It may be concluded that in heavyweight methodology ’Waterfall Model’ & in lightweight methodology ’SCRUM Model’ was most frequently preferred by the developers.

*ii) Use of Methodologies for various Applications:* The developers were asked to register their choices on methodology with respect to the area of application in which the software needs to be developed.

Table 2: Use of Methodologies w.r.t. application areas
(Figures in Percentage)

| Models \ App | Web App | Desktop App | Mobile App | Embedded App | None |
|---|---|---|---|---|---|
| **Heavyweight** | 59 | 17 | 17 | 5 | 2 |
| **Lightweight** | 59 | 14 | 20 | 3 | 4 |

It is evident from the TABLE 2 and FIGURE 2 that in Web Application area both Heavyweight & Lightweight are equally preferred. While developing Desktop Applications Heavyweight methodologies were preferred over the Lightweight methodologies, and in case of Mobile Application development, developers preferred Lightweight methodologies. Hardly a few of the developers (i.e. 7% only) were using Heavyweight and

Lightweight methodologies for development of Embedded and other Applications.

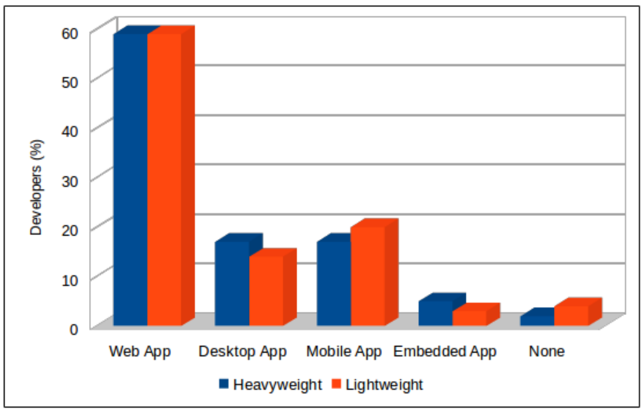

Figure 2: Use of Methodologies w.r.t. application areas

It may be concluded that most of the software are developed in Web Applications and developers were equally preferred both the Lightweight and Heavyweight methodologies.

*iii) Efficiency of Methodologies for Developing the Software*: The developers of different organizations were asked to rate the efficiency of both heavyweight and lightweight methodologies. The efficiency of these models was analyzed by calculating the average score. The average scores for both methodologies are presented in TABLE 3 and FIGIRE 3.

Table 3: Efficiency of Methodologies

(Figures in Percentage)

| Model \ Efficiency | Excellent | Very Good | Good | Poor | Very Poor |
|---|---|---|---|---|---|
| **Heavyweight** | 11 | 43 | 29 | 15 | 2 |
| **Lightweight** | 18 | 50 | 26 | 5 | 1 |

Most of the developers rated the efficiency as 'Very Good' and 'Good' for both lightweight and heavyweight methodologies. Almost similar number of developers responded that both the methodologies are efficient on their own way. Hardly, a few of the developers stated that the efficiency of lightweight methodologies is poor, whereas more than 15% developers opined for poor efficiency of heavyweight methodologies.

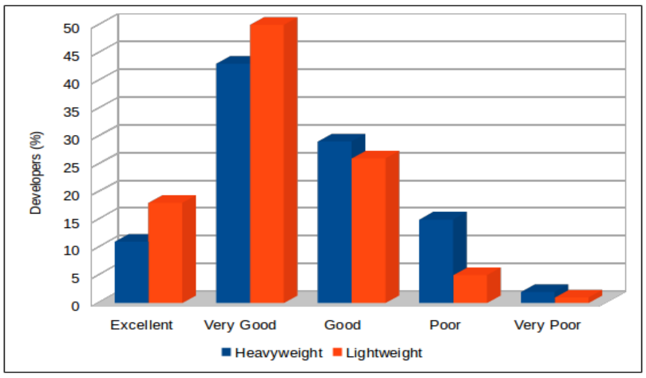

Figure 3: Efficiency of Methodologies

The above discussion lead us to conclude that the lightweight methodologies are more efficient than the heavyweight methodologies.

*iv) Impact of Models on Software Quality*: 'Very Good' to 'Good' quality of software were produced by Heavyweight and Lightweight models. Majority of the developers (36%) were of the view that a Good quality of software were developed with the help of Heavyweight methodology, whereas maximum number of developers (50%) opined that a good quality of software is produced by the Lightweight methodology followed by average quality in both of the cases, as evident from TABLE 4 and FIGURE 4.

Table 4: Impact of Models on Software Quality

(Figures in Percentage)

| Model \ Quality | Very Good | Good | Average | Low | Very Low |
|---|---|---|---|---|---|
| **Heavyweight** | 20 | 36 | 32 | 6 | 6 |
| **Lightweight** | 18 | 50 | 24 | 7 | 1 |

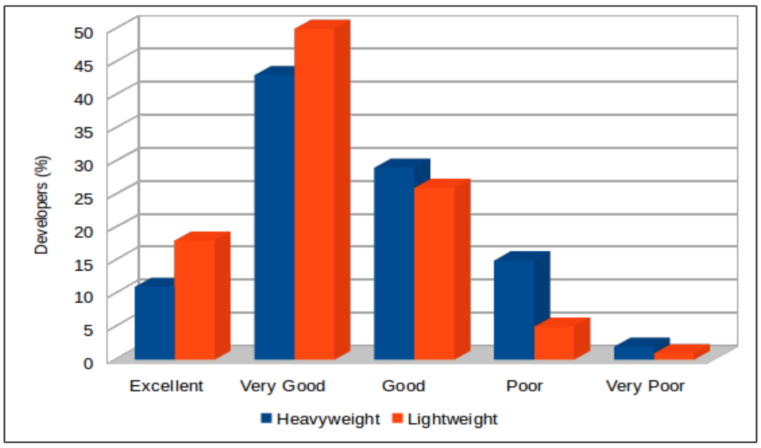

Figure 4: Impact of Models on Software Quality

So, it can be concluded that Heavyweight and Lightweight methodologies both produced a good quality of software. But the inclination of the developers were more towards Lightweight methodology.

**VI CONCLUSION:** There exists various software development models. It is the choice of developer that which methodology is to be preferred over the other for the development of a software project. The Heavyweight and Lightweight methodologies were analyzed and compared on the basis of responses recorded from the developers on the issues like: the use of software development models, the use of methodologies for various application areas, efficiency of methodologies for developing the software and the impact of models on software quality. On the basis of results, it has been found that in Heavyweight methodology 'Waterfall' model and in Lightweight methodology 'SCRUM' was most frequently preferred by the developers. The developers are equally preferring both the Lightweight and Heavyweight methodologies for developing the various applications, whereas the Lightweight methodologies are more efficient than the Heavyweight methodologies. The Heavyweight and Lightweight methodologies both produced a good quality of software.

**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.).

Leau, Y. B., Loo, W. K., Tham, W., Yip, T., & Soo, F. (2012). Software Development Life Cycle AGILE vs Traditional Approaches. *International Conference on Information and Network Technology (ICINT)*, *37*, 162–167.

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

Pawar, R., & Kumar, P. (2015). A Comparative Study of Agile Software Development Methodology and Traditional Waterfall Model. *IOSR Journal of Computer Engineering (IOSR-JCE)*, *01*(08). www.iosrjournals.org

Popli, R., & Chauhan, A. (2013). A Mapping Model for Transforming Traditional Software Development Methods to Agile Methodology. *International Journal of Software Engineering & Applications (IJSEA)*, *4*(4).

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