International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 11 | Nov -2016 www.irjet.net p-ISSN: 2395-0072

LIGHTWEIGHT FRAMEWORK FOR DESCRIBING SOFTWARE BEST PRACTICES

Joyanto Sarkar ,Vivek Sharma, Manjula R

School of Computer Science and Engineering Vellore Institute of Technology, Vellore-632014, Tamil Nadu, India sarkar.joyanto@yahoo.com, sharma.vivek2017@yahoo.com, rmanjula@vit.ac.in

---

ABSTRACT:

With a specific end goal to maximize software project results, software organizations adapt improvement methodologies and implement them in a way that is proper for the project context. This proposes 'Best Practice' is connection subordinate. To better comprehend the logical way of best practice, we require investigating how organizations really approach accomplishing software goals. We require an exploration structure that catches this data in a way that makes no suppositions about practices and that is graphic in nature. Extreme programming and other so-called light-footed or lightweight techniques guarantee to speed and rearrange applications improvement. We have built up a framework in light of the viewpoint that practices exist to meet particular objectives. The catch of hierarchical practices uncovered fascinating instruments for further study, including a reliance upon casual practices connected with solid correspondence and the idea of "push" of data rather than "pull" for software information elicitation.

LIGHTWEIGHT SOFTWARE DEVELOPMENT STRATEGIES:

The lightweight methodology portrays an arrangement of standards for programming improvement under which necessities and arrangements advance through the shared exertion of self-sorting out cross-useful groups. It advances versatile arranging, transformative advancement, early conveyance, and ceaseless change, and it supports quick and adaptable reaction to change. These standards bolster the definition and proceeding with an advancement of numerous product improvement techniques. 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, they are simple, development oriented. They focus more on the participation of people.

RELATED WORKS:

Various exact studies have been done that look to comprehend the operation of programming improvement associations. A number of these studies have as their primary objective assessing or evaluating the association's operation against a specific model then again set of practices. One author tried to determine why software SMEs do not appear to be using standard process models. He interviewed 15 CTOs from SMEs in Ireland and reported their reasons for not using standard models. His main finding was that this was due to the perceived overhead of such process models. He reports why they do not use some models rather than what they do to be successful. There have been a few studies assessing diverse parts of SPI programs. For instance, Cater-Steel, Toleman, and Rout present the consequences of an investigation of 22 Australian SMEs assessed utilizing an evaluation strategy in light of ISO/IEC 15504 (Cater-Steel et al., 2004). Their objective was to decide the degree to which these companies meet the standard and to give an exhortation on programming process change. They distinguished three normal 'issues', one of which was "equipped staff was depended on upon instead of documented forms." Staples et al. did a study to distinguish reasons why associations don't receive CMMI (Staples et al., 2007). Their outcomes showed an observation that CMMI was not for "little" associations, what's more, an observation that CMMI was too immoderate to embrace. Another classification of studies looks at programming improvement rehearses on a district particular premise. In any case, they likewise evaluate against particular programming improvement models or sets of practices. This is valid for the Cusumano et al. (2003) and Cater-Steel et al. (2004) ponders said above. Different illustrations include: Dutta et al. (1999) studied 397 gatherings in 20 European nations to stop mine how many formal models, for example, SPICE and CMMI are utilized; Sison et al. (2006) studied associations in five ASEAN nations (Malaysia, Philippines, Singapore, Thailand and Vietnam) to better comprehend rehearses utilizes as a part of that district. Their review was in light of distributed best-practice, for example, SEI's SWCMM Maturity Survey for Level 2. Our study is particular to New Zealand, however, that was a result of restricted assets instead of a particular objective. The studies portrayed above are significant to comprehend what issues might be connected with specific models, be that as it may, they do not specifically help us comprehend why those associations that don't seem to utilize particular models by and by likewise seem fruitful.

COMPARISON HEAVYWEIGHT V LIGHTWEIGHT:

Heavyweight Strategies:

- High Budget allocation is done.
- Large team size.
- Extremely Critical.
- Process Oriented.
- Explicit knowledge is required.
- Heavy training is required as the software is delivered once it is totally ready.
- More emphasis on process hence no communication.
- Traditional tools and techniques like waterfall model are used.

•Water fall Model.
•Predictive.

Lightweight Strategies:

•Small team size Creative team.
•Low Criticality.
•People Oriented.
•Face to face communication is possible.
•As the development team and the customer are interacting with each other less training is required.
•Face – to – face conversations between the client and the team.
•New techniques like XP, SCRUM management are used. •XP, SCRUM.
•Adaptive Strengths of Lightweight strategies: •Financially savvy.
•Efficiency with lesser group size.
•Lesser documentation required.
•Speedy advancement brings about sparing of time and in addition cost.
•Accentuation on great group union.
•Concentrate on group correspondences, Team soul, and solidarity.
•Test based way to deal with necessities and quality confirmation.
•It gives an open discussion, where everybody knows who is in charge of which thing.

LEADING AGILE DEVELOPMENT METHODOLOGIES

Methodology	Description
Extreme programming (XP)	A lightweight process targeted at development projects that are ill understood or have rapidly changing requirements (Beck, 1999). Enables engineers to confidently react to changing client necessities, even late in the life cycle, stressing collaboration Managers, clients, and designers are all a player in a group devoted to conveying quality programming. Enhances a product venture advancement process in four vital ways: communication, simplicity, feedback, and courage Its development process has various unique features such as requirements as stories, pair programming, test-driven design, frequent unit testing, and continuous integration Primary activities in each XP iteration cycle include new design, error fix, and refactoring.
Adaptive Software Development (ASD)	Places emphasis on production of high-value results emanating from the rapid adaptation to both external and internal events, rather than on the optimization of process improvement techniques. Target development groups in which rivalry makes compelling weight on the conveyance procedure (both high-speed and high-change) on the delivery process
Feature-Driven Development (FDD)	Exceptionally sees little squares of customer esteemed usefulness, called features/highlights, sorting out them into business-related groupings. Concentrates on creating working results at regular intervals and encouraging reviews; administrators recognize what to arrange and how to build up significant milestones. Decreases hazard by underlining the successive conveyance of unmistakable working results. .Provides for detailed planning and measurement guidance; promotes concurrent development within each increment Its motto is “design by feature, build by feature” (Coad, LeFebvre, & Luca, 2000).

Dynamic System Development Method (DSDM)	Dynamic systems development method (DSDM) is an agile project delivery framework principally utilized as a product development technique. Initially discharged in 1994, DSDM initially looked to give some train to the rapid application development (RAD) technique.
Scrum	A team-based approach for controlling the chaos of conflicting interests and needs to iteratively, incrementally develop systems and products when requirements are rapidly changing Can improve communications and maximize cooperation Is scalable from small single projects to entire organizations (Rising & Janoff, 2000)

WHY LIGHTWEIGHT FRAMEWORK?

Maybe the greatest offering purpose of the spry procedures is their weight—or scarcity in that department. There is basically far less to them than to the main heavyweights, which implies conceivably less to learn also, ace. That does not imply that light-footed procedures are simple, any more than it implies that the coordinated procedure can substitute for programming abilities and formative control—however it does imply that there is less to clarify the procedures themselves. Decreased overhead is another solid point. Not at all like the all the more overwhelming obligation forms that stress various deliverables and various procedure ancient rarities, have the dexterous procedures concentrated on code. The configuration is on-the-fly and as required. List cards and whiteboard outlines have the spot of a huge number of outline records, and brief, stand-up confabs supplant extended gatherings. Early results are yet another purpose of an advance of coordinated procedures. With short discharge cycles that produce a completely useful framework on each cycle, nimble strategies empower customers to start utilizing an improved working centre with constrained yet helpful capacity right on time in a task. For customers, directors, and designers alike, a noteworthy potential result originates from the routes in which dexterous strategies diminish the deformity infusion and spillage rates: the quantity of bugs made in any case and the number that sneak past progressive periods of advancement. More the dependable code implies more up-time, less expensive advancement, and less backing and support. The straightforward methods of insight of the coordinated procedures seem to make an interpretation of genuinely well into practice. In spite of—or in view of—the streamlining, coordinated procedures appear to have functioned admirably for an assortment of genuine tasks. The examples of overcoming adversity far dwarf cry stories, at minimum for very much prepared groups taking a shot at properly constrained undertakings.

THE USE:

The bottom–up approach bases what is to be performed as far as improvements on an exhaustive comprehension of the present circumstance. A notable case is quality improvement paradigm (QIP), which proposes a fitting of arrangements taking into account basic issues recognized in the project association. The arrangements are along these lines assessed in pilot projects before an official change is made all the while. The thought is to base improvements on encounters from executing forms in projects, i.e. there is no broad beginning evaluation or examination with a precharacterized set of practices. Rather quantifiable objectives are set and, in view of these, improvements are picked, which can be as new procedures, techniques, procedures or instruments. There been a lot of different opinions on the use of lightweight frameworks and debate among the Members of agile process community. XP and the other light techniques are light on the client side of programming. They appear to be taking care of business in applications that are not GUI-escalated. UI outline, what's more, ease of use is to a great extent neglected by the lithe procedures. With the conceivable exemption of DSDM and FDD, clients and UIs are everything except disregarded out and out. On the whole reasonableness, disregard of clients and UI configuration are fizzling imparted also to the majority of the huge sibling behemoths in the heavyweight field. Some nimble techniques, as XP, do expressly accommodate client or customer interest in sticking down starting necessities through mutually created situations, known as client stories. It is significant that client stories are normally composed of clients or client delegates, not as a matter of course by veritable direct end-clients. Anybody acquainted with utilization - focused configuration comprehends the significance of this qualification. Customers honestly settle on choices in regards to extension and abilities, yet somewhat often don't generally comprehend client needs. Everyone in the field of software is quite familiar that Testing of UIs is serious and tedious work. Genuine ease of use testing requires rehashed testing with quantities of clients under controlled settings. Client or customer responses to paper models are no substitute and can even be totally deceptive. The most basic weakness of all strategies taking into account iterative extension and refinement in little augmentations are the nonappearance of any farreaching outline of the whole design. For inside components of the product, this inadequacy is not deadly, in light of the fact that the design can be refined and rebuilt at a later time. Refactoring can, by and large, compensate for the nonattendance of a complete configuration ahead of time. UIs are an alternate story. With regards to the UI, later

refinement of the design is not a satisfactory choice since it implies changing the framework for clients who have officially learned or aced a before interface. Indeed, even little alterations in the arrangement on the other hand type of components can be risky for clients. Though refactoring of the inner part structure need not as a matter, of course, have any signs on the UI, overhauling the UI engineering is unavoidably problematic for clients. Therefore, the client the interface is one part of the framework that totally should be composed, outlined totally. Iterative prototyping is an adequate substitute for careful UI outline just when the issues are not very confounded when there are not very numerous screens with an excessive number of nuances, and where a fair person on foot and deadened arrangement will suffice. Programming with mind boggling UI issues or for which convenience will be a central point in achievement request a more modern, model-driven way to deal with UI outline. This is the place utilization - focused configuration enters the photo.

CONCLUSION:

From the arrangement of examples of overcoming adversity and recounted proof we have come to trust that supple improvement characterizes a key ability, a capacity to make and react to change, a capacity to adjust adaptability and structure, a capacity to draw imagination and advancement out of an advancement group, and an ability to lead associations through turbulence and vulnerability. The heavyweight arrangement is driven philosophies have an unmistakable spot for a less unstable period, where thorough procedures are material for an extensive variety of tasks. However in this unpredictable environment and expanding the vulnerability of what the client needs, coordinated strategies appear to be the prevailing strategy. In any case, an organization needs to consider which dexterous strategies are advantageous for the organization or some particular activities. The organization can choose distinctive coordinated techniques for various activities or simply tweak any elements that match the state of the organization and the projects. Companies need to make change for their rivals and react rapidly to economic situations. They arrange however are not blinded by those arrangements. They concentrate on conveying client esteem, not including what number of procedures they have set up. They unpleasant out outlines (models) however focus on making working programming. They concentrate on people and their attitudes and on the extreme association of improvement colleagues among themselves, clients and administration. The fate of dexterous techniques appears to be extremely predominant. With regards to techniques, every venture is distinctive. Moreover, it has been obviously seen that alleged nimble philosophies which are versatile to change, individuals situated, fast and responsive are appropriate to the product advancement ventures. Notwithstanding, the organization needs to consider which dexterous techniques are valuable for the organization or some particular undertakings. The organization can choose distinctive coordinated techniques for various ventures or simply redo any components that match the state of the organization and the activities.

SOURCES:

- http://www.umsl.edu/~sauterv/analysis/Fall2013Papers/Egbuchiri/Page7.html
- http://www.agilesherpa.org/intro_to_agile/a_brief_history_of_agile/
- Journal of Systems and Software Volume 85, Issue 6, June 2012, Pages 1213–1221 Special Issue: Agile Development
- Volume 3, Issue 2, February 2013 ISSN: 2277 128XInternational Journal of Advanced Research in Computer Science and Software Engineering
- Adapting Scrum to Managing a Research Group September 18, 2010 Department of Computer Science Technical Report #CS-TR-4966
- K. Schwaber, M. Beedle, Agile Software Development with Scrum, Prentice Hall, Upper Saddle River 2001
- Hansen M.T., Baggesen H., 2009, "From CMMI and Isolation to Scrum, Agile, Lean and collaboration" In Proceedings of Agile, pp. 283- 288.
- http://www.agilealliance.org/
- Cusumano, M., & Yoffie, D. (1999). Software Development on Internet Time. IEEE Computer, 32(10), 60–69.
- Rising, L., & Janoff, N. (2000). The Scrum Software Development Process for Small Teams. IEEE Software, 17(4), 26– 32.
- Coad, P., LeFebvre, E., & Luca, J. D. (2000). Java Modeling in Color with UML: Enterprise Components and Process: Prentice Hall.
- http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0102270
- http://www.ijais.org/archives/volume7/number10/688-1247
- International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868 Foundation of Computer Science FCS, New York, USA Volume 7– No.10, October 2014 – www.ijais.orgby Asma Al Balushi and Santhosh John.