Journal Homepage: - www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH (IJAR)

Article DOI: 10.21474/IJAR01/14795
DOI URL: http://dx.doi.org/10.21474/IJAR01/14795

RESEARCH ARTICLE

A SURVEY REPORT ON PROJECTOMATE

Dr. Manjiri Pathak
Assistant Professor, Department Of Computer Application, Prerna College Of Commerce, Nagpur.

Manuscript Info

Manuscript History
Received: 27 March 2022
Final Accepted: 30 April 2022
Published: May 2022

Key words:-
Agile Development, Agile Project Management, Software Development

Abstract

The Agile Development approach to systems development is the new fundamental for developers. Twelve Principles supporting the Agile Manifesto were elaborated in support. Hence the team always has to look towards identifying and proposing appropriate reference disciplines which are manifest in the Agile Development Universe of Discourse. The Manifesto evokes some reference disciplines to support its various aspects. Why is it desirable and necessary to search for such reference disciplines? The answer is Alternative Software Development and Project Management approaches, such as Software Prototyping and Rapid Development and now including Agile Development, all seemed based on pragmatic experience and have been designed mostly by consultants and practitioners. While this is not in any way to denigrate or deprecate them, it is felt that support for these approaches needs to also come from other disciplines, such as management studies, leadership studies, and the like to give Agile Development a more solid theoretical and intellectual basis.

Introduction:-

Waterfall Model:-

The waterfall model is a breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialization of tasks. The approach is typical for certain areas of engineering design. In software development, it tends to be among the less iterative and flexible approaches, as progress flows in largely one direction ("downwards" like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, deployment, and maintenance.

The waterfall development model originated in the manufacturing and construction industries; where the highly structured physical environments meant that design changes became prohibitively expensive much sooner in the development process. When first adopted for software development, there were no recognized alternatives for knowledge-based creative work.

Agile Development:-

In 2001 what is known as the Agile Development Manifesto was published by a group of consultants and practitioners. The Manifesto was based on extensive experience and knowledge of software development practices and processes. The dimensions of the Agile Development Manifesto and the more explanatory Twelve Principles supporting the Manifesto were counter-intuitive to the then widely accepted Waterfall Approach to software development.

Corresponding Author:- Dr. Manjiri Pathak
Address:- Assistant Professor, Department Of Computer Application, Prerna College Of Commerce, Nagpur.

The thinking behind the Agile Manifesto attempted to identify and differently define the true nature of the software development process. Such definition of its characteristics is missing from the literature. The names in use imply an activity akin to civil engineering and construction projects. Principle among these is the set of practices termed Software Engineering. In Object Oriented development, the Pattern oriented approach includes a Pattern of'software factory'. We see discussion about the role of the Database Architect, a primarily construction role and activity. Project planners are exhorted to 'plan thework and work the plan', and requirements documents are referred to as blueprints. A professional project management activity is the development of a Gantt chart, and students of Project Management inevitably are taught to use some form of project management tool, such as Microsoft Project(R) The Waterfall Model of development, first described by Royce clearly seeks to achieve certainty in the future project activities and certainty in outcomes. The Waterfall Model was originally formalised by authors including Tom deMarco, Edward Yourdon and Larry Constantine and Chris Gane and Trish Sarsonunder the name of Structured Design, or Structured Analysis and Design. It is incommon use still, even though these principles were published in the mid- to late- 1970's. This was during a time when most system being developed could be described as 'batch systems replacing manual clerical systems'.

The 12 Agile Principles:-
It all began back in 2001 with the Agile Manifesto. There was a need for a new approach that can help organizations be more flexible, responsive, and adaptive to changes.

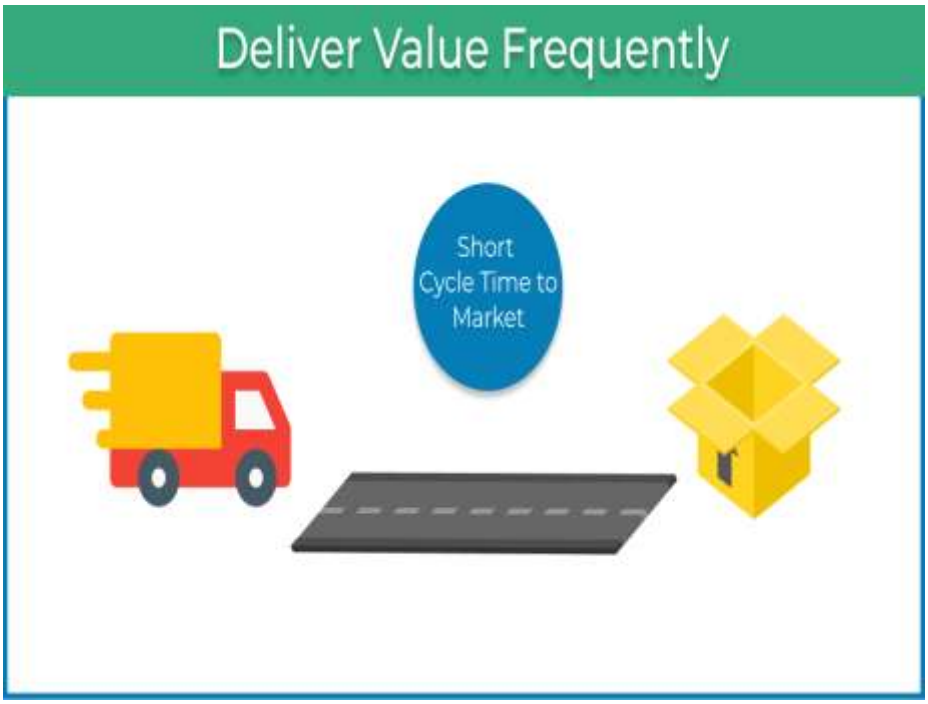
Satisfy Customers Through Early & Continuous Delivery:-
The original formulation of the first of the Agile principles says, "**our highest priority is to satisfy the customer through early and continuous delivery of valuable software**". However, it is perfectly applicable in areas outside of software development.

As you can see, customer satisfaction sits on top of the 12 principles. Early and continuous delivery increases the likelihood of meeting customers' demands and contributes to the generation of faster ROI. By applying this concept, you will increase your process's agility and respond to changes in a timely fashion. On the other hand, your customers will be happier because they will get the value they are paying for more frequently. Also, they will be able to provide you with feedback early on, so you will be able to decrease the likelihood of making significant changes later in the process.

Welcome Changing Requirements Even Late in the Project:-
Still, if need be, change requests should be most welcome even at the late stages of project execution. The original text of the second of the Agile principles states that your team needs to **"welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage"**.

In traditional project management, any late-stage changes are taken with a grain of salt as this usually means scope creep and thus higher costs. In Agile, however, teams aim to embrace uncertainty and acknowledge that even a late change can still bear a lot of value to the end customer. Due to the nature of Agile's iterative process, teams shouldn't have a problem responding to those changes in a timely fashion

Deliver Value Frequently:-
The third Agile project management principle originally states, **"deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale"**. Its prime goal is to reduce the batch sizes that you use to process work.



This principle became necessary due to the extensive amounts of documentation that were part of the planning process in software development at the end of the 20th century. Logically, by taking it to heart, you will reduce the time frame for which you are planning and spend more time working on your projects. In other words, your team will be able to plan in a more agile way.

Break the Silos of Your Project:-

Agile relies on cross-functional teams to make communication easier between the different stakeholders in the project. As the original text states, **"business people and developers must work together daily throughout the project".**

In a knowledge work context that is not explicitly related to software development, you can easily change the word "developers" to "engineers" or "designers" or whatever best suits your situation. The goal is to create a synchronization between the people who create value and those who plan or sell it. This way, you can make internal collaboration seamless and improve your process performance.

Build Projects Around Motivated Individuals:-

The logic behind the fifth of the Agile principles is that by reducing micromanagement and empowering motivated team members, projects will be completed faster and with better quality.

Like the original text following the Agile manifesto states, you need to "**build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done".** The second sentence of this principle is especially important. If you don't trust your team and keep even the tiniest decisions in your company centralized, you will only hinder your team's engagement. As a result, individuals will never feel a sense of belonging to the purpose that a given project is trying to fulfill, and you won't get the most of your potential.

The Most Effective Way of Communication is Face-to-face;-

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

In 2001, this principle was spot on. By communicating in person, you reduce the time between asking a question and receiving an answer. However, in the modern work environment where teams collaborate across the globe, it provides a severe limitation.

Working Software is the Primary Measure of Progress:-
The 7th of the Agile core principles is pretty straightforward. It doesn't matter how many working hours you've invested in your project, how many bugs you managed to fix, or how many lines of code your team has written. If the result of your work is not the way your customer expects it to be, you are in trouble

Maintain a Sustainable Working Pace:-
The precise formulation of this principle is **"Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely."**

Logically, when putting Agile to practice, your goal is to avoid overburden and optimize the way you work so you can frequently deliver to the market and respond to change without requiring personal heroics from your team.

Continuous Excellence Enhances Agility:-
As stated by the Agile Manifesto founders, **"continuous attention to technical excellence and good design enhances agility"**. In a development context, this principle allows teams to create not just working software but also a stable product of high quality.

As a result, changes to the code will be less likely to impact bugs and malfunctions negatively. Still, the 9th of the Agile management principles is applicable in every industry. When you maintain operational excellence, you will have less trouble reacting to changes and maintaining agility.

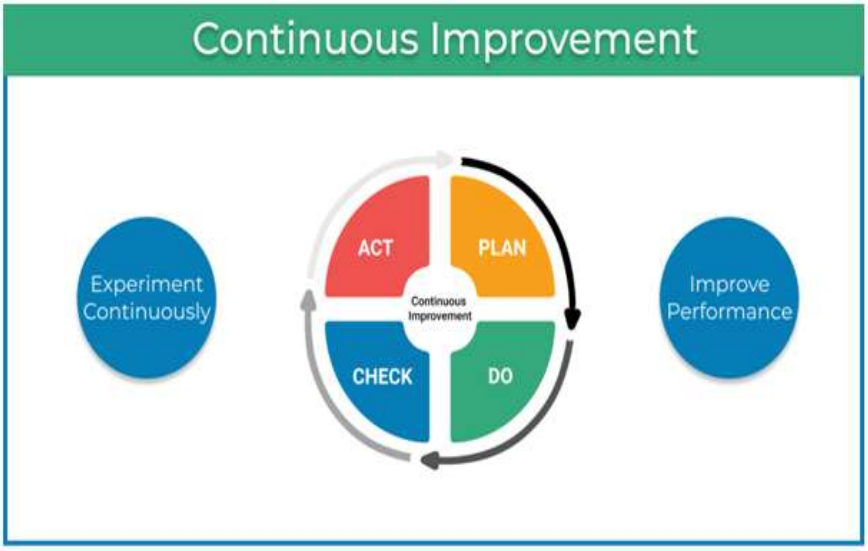
Simplicity is Essential:-
This principle's original content can be a bit confusing as it states, **"Simplicity–the art of maximizing the amount of work not done–is essential"**. Yet, it is very practical.

If you can do something in a simple way, why waste time complicating it? Your customers are not paying for the amount of effort you invest. They are buying a solution to a specific problem that they have. Keep that in mind, when implementing Agile and avoid doing something just for the sake of doing it.

Self-organizing Teams Generate Most Value:-
Once again, we realize that when provided with freedom, motivated teams generate the most value for the customer. When discussing this principle, the 17 fathers of Agile stated that **"the best architectures, requirements, and designs emerge from self-organizing teams".** If you have to push your team and "drive them forward", maybe you are not ready for Agile, or you need to make some changes to your leading style.

Regularly Reflect and Adjust Your Way of Work to Boost Effectiveness:-
Finally, we've come to the last of the Agile management principles. It is related to evaluating your performance and identifying room for improvement. The long version of the principle states: **"At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly"**.

There are different Agile methods, but Agile itself is not a methodology or a framework. It is a set of values and principles. This is why it is incredibly flexible and can be applied by different organizations. However, to make a successful transformation, you need to have the necessary foundation. Implementing the 12 Agile principles is precisely how you build it.



Applicability & Relevance to Agile Methodologies:-
Included under the heading of Agile Methodologies for the purpose of this paper are development approaches that have been called Software Prototyping, Rapid Application Development, Iterative Development, and specifically the 'agile' approaches:

1. **People Focused**: Collaborative, Self-Organizing, and Self-Managing Teams.
2. **Empirical and Adaptive**: 'empirical', 'adaptive', 'evolutionary', 'experiential' development, planning, and estimating.
3. **Iterative**: a series of short iterations each of which produces a useable enhancement to the system.
4. **Incrementa**l: delivering increments to the system.
5. **Evolutionary**: requirements in detail are continuously discovered, and are continually evolving.
6. **Emergent**: The characteristics of the system emerge as parts are added.
7. **Adaptive**: adaptive planning and estimating.
8. **Just-in-Time Requirements Elicitation**: Requirements are stated in detail 'just in time to develop them.
9. **Knowledge-Based**: knowledgeable, self-managing team, continual knowledge sharing and learning.
10. **Client Driven**, 'Pull-Based' development: Only develop what is asked for by the Client, and when the Client asks for it. Agile methods emphasize project transparency, continual communication, and collaboration between project partners.

Detailed Description:-
Our basic aim to develop PMT was to create software that would help manage, maintain, and retain older versions and documentation of a project for developer teams.

The main objective was for the teams to retain the older versions of the project, to easily customize their project workflows, issue types, and fields for the board they want and need, to track the progress of how projects are progressing, and to have access to reports with real-time, actionable insights into how their team is performing sprint over sprint.

Some basic purposes to PMT:-
Purpose I: Seamlessly Fulfilling Course Requirements
Prior to any sort of software design, the requirements for the new software must be clearly defined. This entails matching course objectives to software capabilities. The projectomate software had three main capabilities it needed to address related to basic high-level PM concepts:

a. the application of the solid methodology
b. the managing of costs
c. the ability to share information with relevant stakeholders

Purpose II:
Mapping the technical requirements of projectomateTo sidestep operating system issues and of course delivery, it was decided very early that to provide consistent access to project workers and stakeholders, the software would be developed as a cloud application, making it platform-independent. An additional benefit of developing projectomate as a cloud-based application is that collaboration is inherent in the design of the system from the ground up.

The benefits of using PM tools addresses lots of problems whatever the type of organization is, here's how one can fringe benefit using such solutions:

1. **Collaboration in real-time.** All the team members can focus on their own tasks whereas all the project managers can keep track of multiple tasks or projects.
2. **Cost control.** One of the major objectives of PMs is to control and alleviate expenses. Specialized software guarantees accurate estimations and cost projection.
3. **Document sharing.** Team members and project managers should be able to share project papers and work on those simultaneously.
4. **Monitoring.** This is one of the prime objectives of PM tools. With the help of the projectomate, you are able to monitor the workflow, the progress on all stages of work, and provide forecasts by the time when the project will be accomplished.
5. **Decision making.** Specialized built-in analytics instruments help make data-driven decisions and scrutinize the results.
6. **Risk identification.** With projectomate, you will be able to identify what projects are in danger or face the risks of failure and pay due attention to those projects on time.
7. **Customer satisfaction.** The enactment of project management software allows building stronger relationships with the customers. A PM tool makes sure to deliver projects on time and under budget.
8. **Prioritizing.** Determining what is important can be a daunting task for any managerial chief. By building project management software, you will be able to determine top priorities for your company or project and be attentive to the things that can make a difference.

Conclusion:-

It is suggested that computer software is now so ubiquitous in everyday private and commercial life that no discussion on business, business strategies, etc. can ignore matters pertaining to software.

Software development is seen as a choric activity, defying orderliness and certainty, and therefore demands an appropriate approach to the management of that activity. A paradigm of project management that includes elements of what has been termed The Model of Concurrent Perception, the Learning Organization, Leadership, and Team Dynamics, has been discussed. The software project management approach traditionally used, based on, and inherited from civil engineering and construction project management practices are seen to have failed, as has the management model described as 'command and control'. The 'agile' approaches proffered by a growing number of experts in software development and software project management is the antithesis of this command and control style and is well supported from the literature on management styles and practices, and case studies, that never seem to make it into software project management and software engineering curriculum.

Since the old ways like the waterfall model couldn't give us the equal measures of the projectomate as it has cross-platform issues and bug tracking software with advanced project management capabilities and features.

Top features:-

1. Create user stories and issues, plan sprints
2. Distribute tasks across your software team.
3. Prioritize and discuss your team's work
4. Centralize your team communication
5. See real-time reporting on your team's work

Hence the development of PMT to create software that would help manage, maintain, and retain older versions and documentation of a project for developer teams we give you Projectomate.

Refrences:-

1. 3 Petersen K, Wohlin C. Measuring the flow in lean software development. Software Practice and Experience. 2011; 41(9): 975– 996 Wiley Online LibraryWeb of Science®Google Scholar
2. Graph theoretical indicators and refactoring F. Maurer, D. Wells (Eds.), Extreme Programming and Agile Methods, XP/Agile Universe 2003 (2003), pp. 62-72 View PDF CrossRefView Record in Scopus
3. Author co-citation analysis: overview and defense C.L. Borgman (Ed.), Scholarly Communication and Bibliometrics, Sage Publications, Newbury Park (1990), pp. 84-106 View Record in ScopusGoogle Scholar
4. Is designing software different from designing other things? International Journal of Engineering Education, 22 (2006), pp. 540-550 View Record in ScopusGoogle Scholar
5. The role of physical artefacts in agile software development: two complementary perspectives Interacting with Computers, 21 (2009), pp. 108-116 ArticleDownload PDFCrossRefView Record in ScopusGoogle Scholar
6. Can distributed software development be agile? Communications of the ACM, 49 (2006), pp. 41-46 View PDF CrossRefView Record in ScopusGoogle Scholar Siau, 2005[7] Agile software development: adaptive systems principles and best practices Information Systems Management, 23 (2006), pp. 19-30 View PDF CrossRefView Record in ScopusGoogle Scholar
7. Agile modeling, agile software development, and extreme programming Journal of Database Management, 16 (2005), pp. 88-100 View PDF CrossRefView Record in ScopusGoogle Scholar
8. Empirical studies of agile software development: a systematic review Information and Software Technology, 50 (2008), pp. 833-859 ArticleDownload PDFView Record in Scopus Google ScholarMoe et al., 2009.