Asian Journal of Computer Science and Technology
ISSN: 2249-0701 (P) Vol.10 No.2, 2021, pp.25-33

DOI: https://doi.org/10.51983/ajcst-2021.10.2.2920

# A Comparative Analysis of Agile Methods and the Capacity Maturity Model Integration (CMMI)

**Wumi Ajayi**
Software Engineering Department, School of Computing and Engineering Sciences,
Babcock University, Ilishan Remo, Ogun state, Nigeria
E-mail: wumiajayi1@yahoo.com, ajayiw@babcock.edu.ng
(Received 20 June 2021; Revised 16 July 2021; Accepted 10 August 2021; Available online 20 August 2021)

***Abstract*** **- Quality issue is an important aspect of system development which is defined by a range of plausible elements such as correctness, efficiency, scalability, modularity, maintainability, speed etc. To provide better quality of software than the ones offered by the usual traditional approaches of software development process, new methods like agile methodology was introduced. However, since its introduction, there have been keen contentions as to whether agile methodology offers more advantages than a typical traditional methodology or its numerous "flavours"- that is, a methodology which evolved from a typical traditional approach e.g. the Capacity Maturity Model integration - CMMI. Recent works to determine best practices addressed by three Methodologies shows that both agile and the Capacity Maturity Model integration are now widely used methods in system development process. However, they make different but unique contributions to overall project quality control. Here in this work (using relevant previous works as background), analysis of these two great methods- Agile and CMMI is presented by taken into consideration the fundamental components of both methods, their views about quality, perspective from project management and software development, contributions to overall project quality control and concluded on compatibility of the two methods within the same project. Findings from this work and existing data showed that both can be used in same project to get desirable quality.**
***Keywords:*** **Agile, CMMI, Quality, Software Development, Projects Management**

## I. INTRODUCTION

Development process and its relative terms have been around since late 1950s. Along with the complexity of software itself, the development process contributes more making it a much more complex task; but one that must achieved. There are many views and opinions of the software development process.

Reference [23] defined a software development process as one established to deliver unique software based on agreed terms and conditions between the client and the development team while the software development life cycle (SDLC) is presented as a road map. Again, [32] focused on application development tools, software development process or software life circle was defined as a structure imposed on the development of a software product. Reference [16] viewed software process as "a set of activities, methods, transformation and practices which is used to develop and maintain software and the associated products". The methodology used in the process itself defines things such as "what to do, how is to be done, when (which is the sequence of work) and assignment of tasks/ role to the human resources [17]. The early years of software development process were characterized by the basic traditional approaches like waterfall model, prototyping, spiral [2]. However, due to high rate of system development failure, late delivery and budget overruns etc which are caused by the complications of these traditional methodologies, several other models and approaches to software development process have been invented. Based on [30], Agile methodology for instance, emerged to handle Large projects by defying the plan-driven, traditional, process oriented and lifecycle based approaches through its short iterative cycles of process improvement, constant self-examination procedure, collaborative decision making, built-in quick feedback, change and code modification capability throughout the development process [9]. While on the other hand, the CMMI – Capability Maturity Model Integration – which has its roots from CMM (a leading traditional model), is aimed specifically at using the evolutionary approach to help software organizations develop the maturity of their software processes to a discipline one.

The exploratory work of [16] on the "explore 10 different types of software development process model", established that though there exists some other models that aims at ensuring quality; Agile and CMMI seems to be the most contentious approaches these days. Agile advent particularly has caused serious questioning about its superiority over the older traditional models. Hence for the purpose of this analysis, attention shall be focused on comparison of AGILE methodology (precisely eXtreme programming) and the CMMI.

## II. OBJECTIVES OF THE STUDY

1. To determine the strength of both Agile and CMMI in terms of their unique features.
2. To determine the implementation of both Agile and CMMI together on a project.

## III. METHODOLOGY

The methodology used in developing and presentation of this work is through pure and deep analysis of relevant literatures such as journal articles, reports from research and books.

Conclusion was drawn and recommendations were then given based on the analysis done vis-à-vis existing data sources. Findings from this work and existing data showed that both methods (Agile and CMMI) discussed in-here can be used in same project to get desirable quality. In what follows, an overview and the essential component of both methods are presented.

## IV. THE ESSENTIAL COMPONENTS OF AGILE METHODS AND SEI'S CMMI

In the next two sections, analysis of the components peculiar to agile methodology with reference to Extreme Programming and that of CMMI is presented.

### *A. Agile Methodology (eXtreme Programming): The Main Components*

As analysed in [33] Agile methodologies includes: "eXtreme Programming (XP), crystal methods, scrum, dynamic systems development methodology (DSDM), feature-driven development (FDD), lean, Kanban, crystal and pragmatic Programming" amongst others. The most commonly used ones are the Extreme programming, SCRUM and the FDD [36]. Since agile software development lay emphasis on agility in software production based on some defined values and conforms to agile manifesto, most of the discussions here shall be with reference to the Extreme Programming –XP. The choice of XP was based on the fact that the use of agile methods has increased in recent years; with Scrum and XP taking the lead in the list of the flavours and they have been recognized as very effective methods during projects.

In [29], agile extreme programming is portrayed as one of the most used agile methodologies structured to handle emerging and customer's changing requirement along the development process. Further deductions from [18] analysed alongside [28] shows that eXtreme Programming has twelve practices, grouped into four areas which are derived from the best practices of software engineering.

However generally, as stated in [4], [28] and 13], Extreme Programming consists of the following main components.

*1. Collaboration and Communication: This component* is one of the four values listed in agile manifesto. Reference [18] shows how XP is structured to efficiently accommodate customers' active participation , handle customer's changing requirements and foster easy communication of the changing requirements vis-à-vis development process amongst all stakeholders( which includes the project team members). Iterative development which is driven by product features, customer involvement and constant changes.

*2. Simplicity:* Reference [13] explained that XP applies a very simple approach of planning and tracking to either predict or decide the next line of action along the developmental process. The whole software development process is kept simple and clear in all sense, yet effective [2]. A simple opinion on this is that the XP's simplicity coupled with its ability to reduce unnecessary aspects of a software project into minimal must have contributed to its wide usage and quick acceptability into the industry.

*3. Feedbacks*: XP and all agile methodologies give room for continuous feedbacks from customers and product testing right from the beginning; giving the team advantage to make adequate corrections from time to time along the developmental process.

*4. Courage:* By keeping the process simple and having a clear definition of process tasks coupled with the development of software in bits and giving customers the chance for active involvement in the overall process, the programming team developed enough courage to respond quickly to changes in the requirement.

### *B. Components of CMMI*

The CMMI, is seen as a metal model which evolves from thirty four models that are based on CMM its immediate predecessor [39]**.** In another view from [34] basically, the CMMI is a collection of products used to improve processes. It is organized into six maturity levels to help software organizations improve the maturity of their software development process and as these organizations mature, they move through the levels in ascending other [25]. This evolutionary steps is portrayed in a simpler manner by [21] on CMMI from project manager's perspective. The work highlighted the major areas of CMMI development as shown in fig.1

An in-depth study of this categorization by [21] clearly revealed that some of the mentioned components are all embedded in the categorization of [34] (shown in fig. 1). Hence, the followings can be identified as the main components of the CMMI.

*1. Required Components*: These are components that describe what is to be achieved by the organization to satisfy a process area. Under this category we have: Specific and generic goals. Generally, goals are essential or mandatory CMMI component. These goals as explained in [32], are significantly predictable and relatively stable during the project process. The specific goal has to do with a particular process area and aimed at individual criteria that must be implemented to satisfy the process areas. Specific goals are also used in assessment and reviews to cross check levels of satisfaction of the process area. While generic goal

are goals statement which appears in multiple process areas. Though a process area could only have one defined generic goal.

*2. Expected Components:* These are components that guide the implementation or appraisal team. Essentially, these components help to clearly state what an organization should implement to achieve a required component. Under this category, we have the Specific and generic practices. In a further explanation, [39] claimed that although CMMI focused more on activities but it also incorporates a lot of modern and best practices. These practices include iterative lifecycle, excellent estimation and clear definition of what to do in a particular task. Although, it might not state how the task could be done, but it will describe all the activities which are expected to result in achieving the goals.

*3. Informative Components*: These components provide accurate description that aids the understanding of the required and expected components. Particularly, these components help in providing information needed to describe concepts used in the developmental process. According to [34], The Informative components could be presented as:

*a. Note:* These are used to state uses of components and other relevant information about it. These components could also include remarks and comments.

*b. Examples*: These are needed to clarify a concept or describe an activity.

*c. Reference*: According to [34] and [39], references are pointers showing additional information in related process areas to make things easier and clearer. Other Sub components under this category are: Goals and practice note, Goals and practice titles, references, typical work products, sub practices

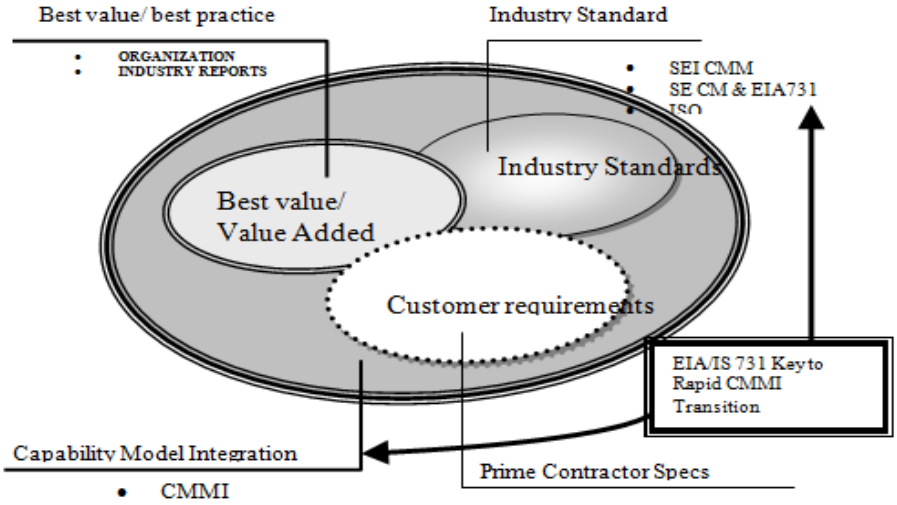
Generally, from the literatures consulted on CMMI so far, it can be deduced that CMMI tends to focus more on activities of the software development and doesn't seems to support the traditional waterfall approaches but tries to incorporate modern best practices as discussed by these authors most especially the SEI's presentations.

## V. CONTRIBUTIONS TO OVERALL PROJECT QUALITY

### *A. CMMI Contributions to Project Quality*

As explained by [31] and [39] CMMI's contribution to overall project quality control includes

1. Defining organizational expectations for a project process.
2. Advices for a scalable and configurable process.
3. Enforces full documentation of all process areas.
4. Reviewing status with high level management.
5. Quick correction of root causes of problems.
6. Specifying that organizations should stick to a strategic and proactive risk management approach.

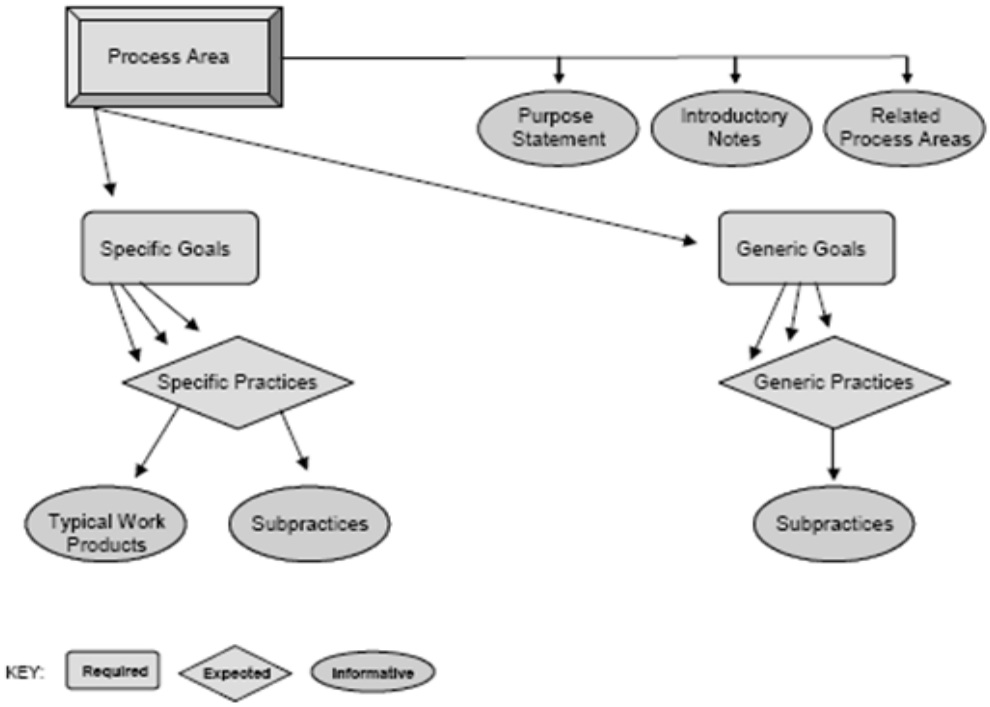



Source [21]

Fig. 1 Evolution of CMMI from other models

### *B. Agile's Contributions to Project Quality*

1. Implementation of good feedback mechanism.
2. Encourages a tactical and reactive approach to risk.
3. Scaling down to meet task size.
4. Implementation of large tasks as series of small projects known as stories (in the XP lexicon) to make room for monitoring, increase efficiency and speed of project delivery.
5. Enforcement of constant changes to meet customer satisfaction.




Source: [34]

Fig. 2 CMMI Model Components

## VI. AGILE METHODOLOGY AGAINST THE CAPACITY MATURITY MODEL INTEGRATION (CMMI)

There have been very many opinions and studies to clearly define the major differences and ascertain the existence of similarities between agile methods and the CMMI in terms of superiority in quality assurance. Obviously the two methods starts from different design point, but based on studies of [19] and [5], both methods are sure concerned with product quality, process management and improvement in the overall process of software development. Nevertheless, there exist major differences in the way product and the process involved in the development of products is viewed. In what follows, a comparison between the duos is presented under specific topics.

### *A. Project Type*

Agile and the SEI CMMI have been found to be useful in different project because of reasons such as project size, the technicality involved or the data orientation, criticality of the project etc. For example, as mentioned in [9] small or medium size projects with low criticality and time sensitive are better handle using agile because of its short iterative cycles, it excellent feedback approach, its average customer involvement, no need for serious planning and simplicity. Such project will not need much of the huge line of activities and documentation like the ones offered in the CMMI for its accomplishments. On the other hand, projects which involves upfront planning, large data, (which means such projects could have some technicalities that requires a relatively stable technology), and because of the technicalities involved vis-à-vis huge effort input (including human efforts), these projects are safety critical; they normally have fixed requirements with few changes. Projects like these can be viewed as large project and because of the size; it would need full documentation so that there would be clear definition of all project areas [34]. Hence, since all these conditions for large projects are catered for in CMMI, it could be thought of as the suitable approach for handling large projects.

### *B. Project Handling Approach*

As explained in [30], [28] and also supported by most of the authors, in a project where agile is being implemented, the project is usually prescriptive and qualitative in approach. Again, processes involved are handled in an incremental and iterative manner. So there is hardly an error gone unnoticed and this gives room for constant self-examination procedure, built-in quick feedback, change, and code modification capability all through the development process. Quick response to change as the project progresses with full attention on end product. The process is usually a light weight one with reduced guidelines and documentation.

The agile development process laid strong emphasis on communication and collaboration. This means both the team and clients communicate and work together to meet the requirements specified by the client. As explained in [25], everybody plays virtually all roles such as developer, designer just to be involved and knows what is going on at a stage or the other. Personally, though the collaboration attribute is a good one because customer's requirement can

be reviewed from time to time. However, this could lead to a serious problem for the project because, the development process may be slower than necessary since customers are going to be involved in direct communication with the team and attention may sometimes be shifted from the original motive of the software due to constant changes in the requirement.

As against agile, In CMMI, full description and quantitative analysis is chosen over prescriptive and qualitative nature of Agile. Rather than communication and collaboration during process handling using the agile methodology, attention is more on contract negotiation and documentation with series of guidelines. Due to huge documentation the process becomes a heavy weight method. Furthermore, as against responding to change, the CMMI follows a drafted plan in an evolutionary approach to ensure that organizations develop in their software processes. Through this, it is easier when CMMI is being implemented to focus more on activities because it is rooted in the belief that if the process is right, then the end result must definitely be alright. This is against the belief of agile methodology which emphasizes more on the product's quality rather than the process that produce the result.

C. *Ways of Ensuring Quality*

A good study of [8], [25] and other relevant materials revealed that quality in most flavors like XP is ensured by the following.

*1. Frequent Inspection*: f*requent inspection* of project processes coupled with continuous process change until customer requirement is met.
*2. Excellent Teamwork*: In XP, Behavior Driven Development (BDD) is practiced which aid the serious adoption of pair programming (aside other roles present within the project.).
The pair programming involves programming in twos or threes to allow room for discussion about meeting requirements design, testing and programming concerns. A major beauty of this is that correction of task or debugging doesn't have to wait for someone, their separate and collective codes could be amended at anytime so long the person to carry out the activity is a member of the coding team.
*3. Simple Design and Small Releases:* the designs are kept simple so that full control over the quality can be maintained and then released into the market as quickly as possible. This gives the product an advantage of being used or tested quickly for subsequent feedbacks which is then modified to a better one to be release at a later date.
*4. Implementation* of standard and consistent coding practices
*5. Onsite Customer and Refactoring*: one of the ways of ensuring quality is making the customer available on site so that they can be involved in the developmental process.
*6. Continuous Integration:* the coding team does not integrate the code ones or twice, codes are continuously rebuilt and retested (in an automated fashion) whenever the need arise for amendments.

Conversely, in CMMI quality is ensured by

*1. Advance Project Planning*: this starts from the initial blue print of the whole project to finish (including the risk area).
*2. Better Best Practices:* unlike the agile methodology (XP), the CMMI incorporate more modern best practices such as explicit feedback loop between requirements development and delivery of product to customer.
*3. Approach:* scaled down approach of predictable and doable tasks carried out with strong process assets.
*4. Strict Adherence to Rules:* The CMMI tries to follow laid down rules and adhere strictly to plans to ensure that product are designed and delivered to specifications.
*5. Clear Definition of Maturity Levels:* CMMI defines maturity levels which organization's product quality must satisfy before moving up to the next higher level. Which means certifications earn for each of these level could be a yardstick for the quality of products expected at the end of the day. Although, judging by the work of [3]**,** one could say that the SEI CMMI sometimes loses its credibility as its much expected productivity and quality improvements do not increase as the maturity levels increase. This is because part of the expectations of CMMI users is that an improvement in the appraisal will improve the productivity and quality of their organizations. However, when this is not so, it is like a hope lost and the methodology is discredited.

*D. Project Management Perspective*

Though agile methodology and CMMI have both been used successfully by project managers for different types of projects, nevertheless their perspective differs looking deeply into their implementations during the course of the project. Going by [39] and [21], the CMMI addresses six project management areas. It is believed in project management that these areas cover all the necessary activities involve in a project and once proper attention is focus on them, they will facilitate or ensure the delivery of a quality product at the end of the day. These areas include:

*1. Project Planning* (which include drawing of the whole project plan itself, taking the estimates and commitment to the drawn plan)
*2. Project Monitoring and Control* which involves monitoring the project against the plan from time to time and managing corrective procedure from start to finish time
*3. Risk Management*: this covers all the risky areas. This means identifying risk factors and risky areas of the project. It also includes preparation for the risk and forestall or make plans for alleviation. Other areas are supplier agreement management which looks into establishment and satisfying of supplier agreements. Finally, Integrated Project Management which manages the stakeholder involvement and Quantitative project management.

As against the project management perspective of the CMMI, [15] established that when agile approach is viewed from project management, the expectation is that a project plan always evolve in response to change as the development progresses. So instead of wasting what is believed to be a valuable time on drawing an initial plan, it concentrate more on the project with high level of customer Involvement and frequent delivery of working software which is used as mini plans. Customer requirements are then documented as reports or stories which serve as basis for estimating and planning**.**

*E. Views about Quality, Quality Measurement and Standard of Measurement*

*1. Views about Quality:* The work of [6] shows that agile methodologies and the CMMI have clearly different and well defined views about quality. While agile places importance on obtaining the smallest workable piece of functionality which delivers business value quickly through continuous improvement, collaboration and communication with customer in adding further functionality throughout the life of the project, the CMMI on the other hand, believes a good process of software development should produce an excellent quality of software. This then allow CMMI to focus all its attention on generating a measurable and improved process through strict adherence to lay down rules, full definition of goals, risk averting, uniformity in process, building of customer trust in process infrastructure, quantitative scientific analysis and documentation of process tasks.

*2. Quality Measurement and Standard of Measurement:* According to [24] Measurement refers to the "process of assigning numbers or symbols to attributes of entities in the real world in such away as to describe them, accordingly to clearly defined rules". Measuring quality can be viewed from several aspects [8] and [28]. One of this is in terms of the different components involved in its makeup or the quality attributes. Again as deduced in [1] and [25], for software to be rated a high quality, it must have satisfy (substantially) all the applicable quality attributes like: economy, correctness, clarity, documentation, efficiency, reusability, understandability etc therefore, judging from all these assertions agile methodology can be said measures quality from these angles:

*3. Economy*: These deals with issues like: is it cost effective? Or is it reusable?

*4. Customer Perspective*: The perspective of the product with reference to portability, satisfaction, acceptability, efficiency and ease of modification to meet customer changing requirement.

*5. Market Value*: This include both negative and positive feedbacks and responses from the market (testability), customer satisfaction, correctness (is the code or product acceptable). Whereas in CMMI, quality is measured by,

*6. Performance:* This involves performance details and understanding of the process.

*7. Efficiency*: has the process been able to produce workable software (efficiency. Stability and maintainability of the technology: is there a proper documentation for it?

*8. Risk:* Is the process totally risk free or less of risks (Less or total risk averting). All these factors cumulate into quality measurement.

Software standard of measurement (most especially size measurement) has been a thing of concern in the software engineering [10]. Without solid baseline for size, resource estimation, planning and control for projects (large or small scale) might seem impossible. Generally, apart from size, other major things to measure as explained by [25] include: schedule and resource usage, cost and reliability. For these purpose, several models have been developed to measure each of these project essentials. For example, the size of a software project could be measured using function points and lines of codes or KLOC.

While schedules and resources (including cost) can be measured using the famous Constructive Cost Model (COCOMO) approach. Other than these models there are certain standard set by standard organizations to measure general level of success of implementation of a methodology in a project. For instance, the Agility Index Measurement also known as AIM, is used in agile methodology to appraise level of its success in a project [43] & [45]. The AIM tries to appraise a project against a number of agility factors before arriving at its conclusion on its success in a project. Again, Agility Measurement Index is used to compare development and five dimensions of a software project (duration, risk, novelty, effort, and interaction).

In CMMI, basically, things to measure include: Schedules and progress, effort and cost, size and stability and of course, quality. The standards for measuring quality as explained by [35] on CMMI overview for Executives are

*i. Appraisals: The CMMI Appraisals (SCAMPI$^{SM}$)* which the official SEI method for providing benchmark-quality ratings. It makes use of the CMMI maturity models as yardsticks against which organizations' level of maturity in product development, acquisition and project processes are measured in respect to the industry standard**.** The SCAMPI appraisals are the main tool used for identifying strengths and weaknesses of current processes, reveals development/ acquisition risks, and determine capability and maturity level ratings. The SCAMPI defines three appraisal classes-A, B and C. and ratings; final reporting and follow-on activities.

*ii. CMMI Best Practices*: This is used by organizations for benchmarking one another in a variety of industries.

*F. Is Agile then a risk to Quality?*

The answer to this question is very subjective depending on the angle of consideration. For example, as shown in fig 3 and [20], if considered from the documentation or process intensiveness, tools usage, product quality etc. agile could not have done much in these regards. While a methodology like CMMI would be a force to reckon with for most projects when it comes to documentation and probably risk analysis. Be that as it may, while some say agility doesn't provide enough rigors to ensure quality, some still say otherwise [38] with a broad opinion that agile is just too perfect for most projects. Generally, it is believed that the perspective of IT manager and project management towards the use of agile in projects tends towards meeting customer satisfaction and deliverables. While, CMMI is viewed as a methodology which tends to benchmark the processes involve in the development of the deliverable (software) with keen interest on risk management, stability in technology and documentation rather than the product itself [21]. However, based on all the analysis presented so far, one might be tempted to go with the opinion of [39] and [21] claiming that although CMMI may be broad, inclusive and difficult but it's always seeing by the practitioner as being a discipline approach with set of rules to follow and could easily avert risks due to its strong contingency plans. Agile on the other hand, could be speedy and creative in nature, but too informal because of its lack of plans and less attention to risk management which may not favour the cost effectiveness factor of quality and also poses a serious threat to quality. Again, it must be noted that agile transformation comes with some challenges such as organization and management, human related and process based [44].


Source: [20]

Fig. 3 Agile Statistics within a Project

*G. Major Similarities of Agile and the CMMI*

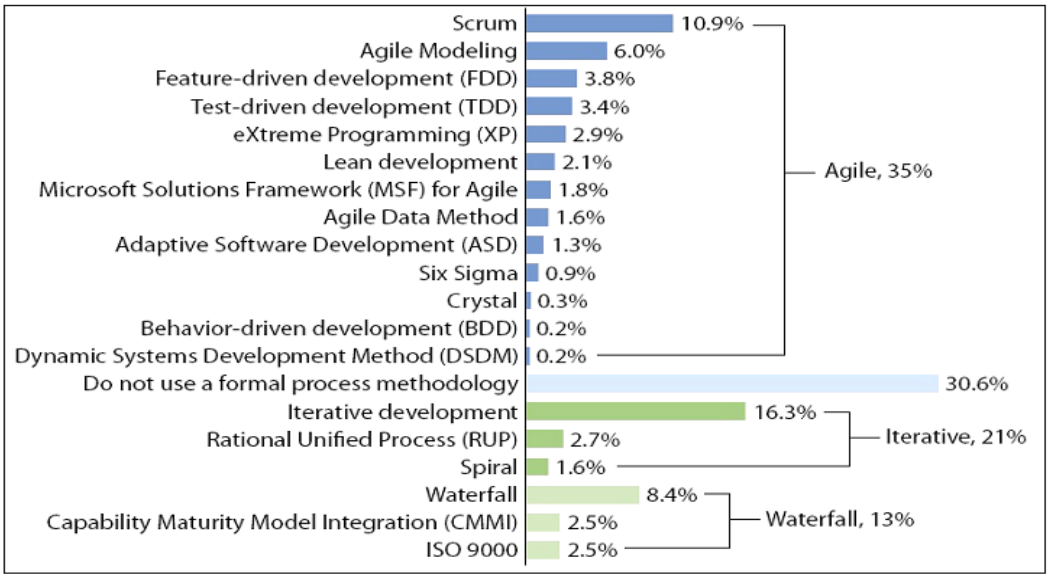
Proponents of agile methodologies and the CMMI have for a long time observed the processes involved in both and considered them to be mutually exclusive. Despite the major differences which their proponent claimed hinders the existence of similarities between the two, research have found that they can be complementary.

Ref [41] and [42] revealed that both Agile and CMMI are governed by rules; they both have defined goals for the process being used for so they must plan to achieve the goals (Although the plan is more pronounced in CMMI).

Ref [14] on understanding the relationship between agile methodologies and other classical approaches further proved that agile methodologies and the CMMI share some major similarities. For example, both XP and CMM (the immediate predecessor of CMMI) are philosophically compatible; this means CMMI shares same fate. Again, in [37] it was shown that XP and some other agile methodologies support about eleven components of CMMI. Although several others of same similarities are still being considered as mere hypothesis. Nevertheless, it must be noted that the two approaches offers their personal value when applied to different software projects [6].

## VII. DISCUSSION AND RECOMMENDATIONS

Having examined critically several works related to the topic addressed herein this analysis, a conclusion can easily be drawn that recommendation for the use of any agile methodologies or not depends on the type and size of project being considered. Ref [7] recommended seven factors you need to adopt for agile projects to be successful. These include but not limited to focusing on people above processes & mind-set above skills, proper integration of testing and the development lifecycle, ensure a self-organizing teams, success redefinition, promote team communication, proper documentation and periodic reviews.




Source: [22]

Fig. 4 Percentages of usage of methods

Also [1] and [6] maintains that the use of agile methods is more suitable when the concern project is a small, time sensitive and a changing requirement of the software project. Agile proponents claim the use of it tends to offer more advantages than the use of CMMI in the development of software projects [38] and [9]. Although some professional are against this; for example some of the criticisms against the use of XP (which is a flavor of agile) is that XP works only with top developers and that it is a way of extorting money from the customer through small releases. Hence, it is advised that the IT industry and the business world which requires the services provided by these two approaches present different problems, if possible at different times. Then users can always employ the service of the suitable one as the project comes; bearing in mind crucial factors like size, time of delivery, technicality, data orientation and environment. For instance, with the leading role of a good agile manager who understands the effects of mutual interactions amongst project areas, agile could be used successfully in an unpredictable project environments rather than the CMMI. This could be achieved by steering all stakeholders in the direction of continuous learning and adaptation as project progresses. Again, in a project that involves too many technicalities and requires stable technology, the full implementation of the CMMI may be deployed.

## VIII. CONCLUSSION

Reports of [22] revealed that there is good indication that agile is now being adopted by several organizations. In the technical report of [11] on 'CMMI or Agile: why Not Embrace both' there are clear indications of compatibility of agile methodologies and CMMI within the same project. Although as proven from [38] that a project considered big or large could engage the approach of CMMI because generally, a project of such magnitude will require fore planning, large number of team members, resources, risk analysis and management so will definitely need huge documentation (with note, references etc.) to make clear each project areas. Although provisions for such strength have now been improved and made available in the newer versions of CMMI [26].

As exemplified in [27] on "Benefits of Blending Agile and Waterfall Planning Methodologies", agile methodologies do very well with small, changing and time sensitive projects too. On a closer look, bigger projects are likely to change more during execution; which means one can easily incorporate some aspect of agile into projects considered big just to strike a good balance in the implementation of the duo where applicable. On a last note, based on previous work such as [27], it is an opinion that agile methodologies and the CMMI are both good methodologies, although with different views about software development process and quality as a whole. It can be said convincingly that the disagreement between proponents of the two methodologies is more about personal and organizational norms and culture which defines how a software development process is best organized and performed.

## REFERENCES

[1] G. Arcidiacono, N. Costantino and K. Yang , "The AMSE Lean Six Sigma governance model", *International Journal of Lean Six Sigma*, 2016. ISSN: 2040-4166. [Online]. Available: https://www.google.com/imgres?imgurl=x-raw-image%3A%2F%2F%2F52a26b9d173f3ed976b18c8382cefaf5031e18e960e291048f8e296558375557&imgrefurl=https%3A%2F%2Fwww.emerald.com

[2] J. Armitage, Are Agile Methods Good for Design? , Interactions, Vol. 11, No. 1, ACM, pp 1-23, 2004.

[3] R. Baker, The Corporate Politics of CMM Ratings Technical Opinion, ACM, West Newbury, MA 1996, Vol. 39, No. 9, pp. 105-106,1996.

[4] Batra, W. Xia and M. Mingyu, "Collaboration in Agile Software Development: Concept and Dimensions", *Communications of the Association for Information Systems*, Vol. 41, pp. 429-449, 2017.

[5] Biberoglu and H. Haddad, "Survey of industrial experiences with CMM and the teaching of CMM practices", *Journal of Computing Sciences in Colleges, CCSC*, Vol. 18, No. 2, pp. 143-152, 2002.

[6] B. W. Boehm, "Get ready for the agile methods, with care", *IEEE Computer*, Vol. 35, No.1, pp. 64-69, Jan 2002.

[7] Cigniti, *7 Recommendations You Need To Adopt For Agile Projects To Be Successful*, 2021, [Online]. Available: https: www.cigniti.com/ blog/7-recommendations-need-adopt-agile-projects-successful.

[8] Coffin and Lane, *A Practical Guide To Seven Agile Methodologies, Part 1*, 2007. [Online]. Available: http://www.devx.com/architect/ Article/32761/1954.

[9] C. Ferreira and J. Cohen, "Agile Systems Development and Stakeholder Satisfaction: A South African Empirical Study", *Proceedings of SAICSIT, ACM, Wilderness South Africa*, pp. 48-55, 2008.

[10] C. Gencel and O. Demirors, *Functional Size Measurement Revisited*, Middle East Technical University, 2008. [Online]. Available: http://doi.acm.org/10.1145/1363102.1363106.

[11] H. Glazer J. Dalton, D. Anderson, M Konrad, and S. Shrum, *CMMI or Agile Why Not Embrace Both!*, SEI technical report, 2008. [Online]. Available: http://www.scribd.com/doc/11026560/CMMI-or-Agile-Why-Not-Embrace-Both.

[12] L. R. Guimarães and P. R. S. Vilela, "Comparing software development models using CDM", *SIGITE '05: Proceedings of the 6th conference on Information technology education, ACM*, Newark, NJ, USA Session, pp. 339-347, 2005.

[13] R. E. Jefreies, 2009. [Online]. Available: http://www.xprogramming. com/xpmag/whatisxp.htm.

[14] L. Jiang and A. Eberlein, "Towards a framework for understanding the relationships between classical software engineering and agile methodologies", *Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral*, Leipzig Germany ACM, pp. 9-14, 2008.

[15] Keenan, S. Powell, G. Coleman and K. Mc Daid, "Learning Project Planning the Agile Way", *ITICSE'06: Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education, ACM*, Italy, pp. 324, 2006.

[16] S.T. Krishna, S. Sreekanth, K. Perumal and K. R. Reddy, "Explore 10 Different Types of Software Development Process Models", *International Journal of Computer Science and Information Technologies*, Vol. 3, No. 4, pp. 4580-4584, 2012.

[17] Lean-Kanban University-[LKU], 2013. [Online]. Available: https://resources.kanban.university/conferencearchive/lkna13.

[18] J. C. Lee, "Embracing Agile Development of Usable Software Systems", *CHI '06: CHI '06 extended abstracts on Human factors in computing systems, ACM*, Québec, Canada, pp.1767-1770, 2006.

[19] K. Madsen, "Agility vs. Stability at a Successful Start-Up: Steps to Progress Amidst Chaos and Change", *OOPSLA '05: Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, ACM*, California, USA, pp. 313-318, 2005.

[20] R. Mantovani Fontana, I. Mantovani Fontana, P. A. R. Garbuio, S. Reinehr and Andreia Malucelli, "Processes versus people: How should agile software development maturity be defined?," *Journal of Systems and Software*, Vol. 97, pp. 140-155, 2014. [Online]. Available: https://www.sciencedirect.com/Science/article/abs/pii/ S0164121214001587.

[21] J. Martak, Capability Maturity Model Integration (CMMI) From a Project Management Perspective, 2004. [Online]. Available: http://www.unicom.co.uk/product_detail.asp?prdid=1477.

[22] A. B. M. Moniruzzaman and S. A. Hossain, "Comparative Study on Agile software development Methodologies", 2013. [Online]. Available: https://www.researchgate.net/publication/249011841_ Comparative_Study_on_Agile_software_developmentmethodologies.

[23] Oleksandrova, 2018. [Online]. Available: https://codeit.us/blog/ software_development-life-cycle.

[24] R. K. Pandey, "Relativity in Software Engineering Measurements", *SIGSOFT Software Engineering Notes, ACM*, Vol. 34, No. 2, pp. 1-3, March 2009.

[25] M. Pearson, *Lecture Notes and Slides for MSc Information Technology Management*, 2008/2009 Session, University of Sunderland, United Kingdom, 2009.

[26] M. Phillips and S. Shrum, "Process Improvement for All: What to Expect from CMMI Version 1.3", *CROSSTALK The Journal of Defense Software Engineering*, Software Engineering Institute, pp. 10-15, 2010.

[27] C. Quist, "Benefits of Blending Agile and Waterfall Project Planning Methodologies", *Applied Information Management Master's Capstone Projects and Papers [215]*, 2015. [Online]. Available: http://hdl.handle.net/1794/19634

[28] W. Radinger and K. M. Goeschka, "Agile Software Development for Component Based Software Engineering", *OOPSLA '03: Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, ACM*, California, USA, pp. 300-301, 2003.

[29] M. W. Raja and K. Nirmala," An extreme programming method for e-learning course for web application development", *International Journal of Engineering Sciences & Research Technology*, pp. 560-569, 2016. [Online]. Available: http://www.ijesrt.com/issues%20 pdf%20file/Archive-2016/April2016/81_AN%20extreme%20 programming%20method%20for%20elearning%20course%20for%2 0web%20application%20development.pdf

[30] N. Rashid, "Applying Agile Methodologies on Large Software Projects", *International Journal of Recent Research in Mathematics Computer Science and Information Technology*, Vol. 2, No. 1, pp. 273-278, 2015. [Online]. Available: www.paperpublications.org

[31] D. J. Reifer, F. Maurer and H. Erdogmus, Aling Agile Methods Software, *IEEE*, Vol. 20, No. 4, pp. 12-14, July-Aug. 2003.

[32] SBS, 2010, [Online]. Available: http://www.selectbs.com/adt/ analysis-anddesign/what-is-a-software-development-process on 7th march, 2010.

[33] S. Sharma, D. Sarkar and D. Gupta, "Agile Processes and Methodologies: A Conceptual Study", *International Journal on Computer Science and Engineering (IJCSE)*, Vol. 4, No. 5, pp. 892-898, 2012.

[34] SEI Software Engineering Institute, 2010, [Online] available: http://www.sei.cmu.edu/cmmi/general.

[35] S. Shrum and M. Phillips, "Process Improvement for all: What to expect from CMMI version 1.3", *The Journal of Defense Software Engineering*, pp. 10-15, 2010. [Online]. Available: http://www.cs.cmu.edu/~bam/uicourse/2011hasd/Phillips%202010% 20%20What%20to%20Expect%20from%20CMMI%20Version%201 .3%20(Crosstalk).pdf.

[36] C. Tan, W. Tan and H. Teo, "Training Students to be Agile Information Systems Developers: A Pedagogical Approach", *SIGMIS-CPR '08: Proceedings of the 2008 ACM SIGMIS CPR conference on Computer personnel doctoral consortium and research, ACM*, Virginia, USA pp. 88-96, 2008.

[37] R. Turner and A. Jain, "Agile Meets CMMI: Culture Clash or Common Cause?", *In Wells D. and Williams L. (Eds.): LNCS, XP/Agile Universe 2002*, pp. 153-165, 2002.

[38] W. H. M. Theunissen, A. Boake, and D. G. Kourie, "In search of the sweet spot: agile open collaborative corporate software development", *SAICSIT '05: Proceedings of the 2005 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries, SAICSIT Journal*, pp. 268-277, 2005.

[39] T. Veldhuizen, *CMMI & Process Improvement (evolution of CMMI)*, pp. 1-3, 2007. [Online]. Available: http://www.sei.cmu.edu/cmmi/ general.

[40] [Online]. Available: http://www.globalworldtech.com/methods.html.

[41] [Online]. Available: http://www.selectbs.com/adt/process-maturity/ what-is-capability-maturity-model-integration.

[42] [Online]. Available: http://www.agilemethod.net/templates/scope-templates.html.

[43] [Online]. Available: https://www.aim.com.au/project-management/ courses/agile-project-management.

[44] [Online]. Available: https://www.researchgate.net/publication/2370 77450_Obstacles_in_moving_to_agile_software_development_metho ds_At_a_Glance.

[45] [Online]. Available: https://www.aim.com.au/project-management/ courses/agile-fundamentals.