**CMMI Process and Agile Methodology for Software Development**

**A practical approach with ASP.NET**

**Introduction:**

Wonder, one more new article on CMMI and Agile. Yes, but this time with a practical example. What makes me write this article even when there are lots of stuff on internet regarding both CMMI and Agile, is many have only theoretical approach, which makes both the topic just read and forget. So I decided to share what I understood with CMMI and Agile by following up a practical example. Making a practical project is quite different as compare to reading. But off course I will start with little practical theory and then follow up with an example project.

**CMMI:**

**What is CMMI?** Capability Maturity Model Integration, off course, but still what is CMMI? By elaborating an abbreviation can we understand CMMI? Or is that the only definition? Obvious answer is NO! So let’s start from ground till up. First it should be clear that CMMI is not a certification. No organization can be certified with CMMI, rather an organization is *appraised* to CMMI. This means there are some set of rules and regulations which CMMI describes and an organization is compared how best the set of rules and regulation are followed. By comparing these rules, CMMI is appraised, simple. Now as some part is clear let’s go up. So before coming to CMMI, just concentrate on CMM, before that on MM. MM is maturity model (please note model is not module, module is a part of model). Let’s take a quick historical example:

We all know (I hope so), that **Benz** Patent Motor Car was the first automobile company for manufacturing cars, they started operating in 1885. Means they made the first model for manufacturing cars. This model was so success that others studied it and started manufacturing their own brands of car! Please don’t concentrate on car, rather concentrate on the MODEL made by the Benz. This MODEL was so mature (by further studying and investigating) that other started to follow to manufacture car. Here the MODEL becomes MATURE MODEL, and the process which makes MODEL to MATURE MODEL is nothing but MATURITY MODEL (MM). Now when we are clear on MM in CMMI, let’s go further with our car’s example.

Making a perfect car was not the end of automobile manufacturing, it went further to make and build the world’s best car! Lexus, a division of Toyota Motor Corporation, introduced the world best car in 1989 in USA. So as to launch the world’s best car Lexus started its plan in 1983 under project code name F1. Stop, don’t get carried away in history, here the “code name”, a “plan” and term “the world best’s car” is simply putting us under the capability to do, not the better, but the best in the industry, which Lexus did! Persons who were involved in this CAPABILITY took the MATURITY MODEL process to CAPABILITY MATURITY MODEL, and till date Toyota Motor upholds this and followed by many.

Further we all know that today’s cars are becoming more advanced in technology, which will make automobile in next generation as hybrid, means today we just imagine that car is running without a driver, while self-driven automobiles are already manufactured! This is an integration of many technologies in automobile, got it! NO? I will explain, here our CAPABILITY MAURITY MODEL now becomes CAPBILITY MATURITY MODEL INTEGRATION!

So in nutshell Capability Maturity Model Integration (CMMI), is simply initializing (making a car), managing (just putting a car to be runnable on road), defining (to define, the best car), quantitatively managing (to achieve the best car) and optimizing (making the best car). Bingo, we are done with CMMI! How?

The five points 1. Initializing 2. Managing 3. Defining 4. Quantitatively Managing 5. Optimizing are nothing but the five levels of CMMI. Let’s have a quick round of how these five points, means how they are measured. Here please note the CMMI’s 5 levels are different for different purposes (acquisition, development and service model) , but as we are talking about CMMI for software purpose, following measures apply to the development process of software application (project) and the organization under which this application is developed, means CMMI for software development model.

1. Any project just starts with the basic idea of what will be the project itself, theoretically it is just one liner which elaborate a project. (This is true, not believing? Just think what is Google? Answer is search engine, right. A one liner project!) On this we start doing modeling means we start creating a project. This can be Level 1 which simply is initializing a project. (Making a software application, *the car*)
2. We all know that practically a project is never a one-liner, hence, the stack holder (for whom the project is made) must have agreement of the service he needs, what all the requirements will be, for that requirements what will be the configurations, how configurations are analyzed and measured, planned, controlled, assured for the quality and the process of configuration, these all managing accepts are under Level 2 of CMMI. (Putting a software application to execute, *just running a car*)
3. As we know, no one is perfect, so is the stack holder (client) and the organization’s resources. Hence there may be changes to change, modify or extend, from time to time, so how requirements are gathered and developed? Does analysis and resolutions are decided? What was the technical solution achieved or proposed for changes to the stack holder? Was the underline risk managed? How all these are integrated and managed with the project? How organization organized these processes and their definitions? What is the organization’s training and induction imparted to their resources? How was the validation and verification achieved? These measures are of Level 3 of CMMI. (defining each accepts to achieve the best, thinking of *the best car*)
4. A project is never under an individual or team or group of team, a project is always a commitment between an organization and a stack holder, hence what are the processes of performance and quantitatively managing the processes of a project(s)? This is measure under Level 4 of CMMI.(process to achieve the best, *achieving the best car*)
5. Doing all the best, may not have the best result, so to insure the best what is the optimization done in form of organizational performance management? How are the casual analysis and resolution carried out? (the best reached but iterate over and confirmed that is best, *making and rolling out the best car*)

I guess many points over CMMI would be cleared till now. For further more elaboration one can go on the official website of CMMI and have more in-depth course. But as for software application or project and understanding this is enough. Please note that CMMI Level 1 is by default, it is not appraise and CMMI appraise starts with Level 2. Now let’s move to the other part.

**Agile Software development.**

**Agile Concept:** Before going into Agile Software Development, let it be very clear that there being no technological related steps in Agile Software Development process. Hence we should first understand what Agile Methodology is. Just recall that while understanding CMMI, we took the example of automobile industry means, the car. Let’s take the same to understand what’s the Agile Concept is. So let’s go again in history. Ready?

So, as we know it was Toyota Motor Company which defined and rolled out the best car. But this didn’t came just overnight and I am saying not in couple of years also. It was based on three basic platforms which where layout long back, let’s get on that now.

1. Many may not know that Toyota was not an automobile industry but was textile industry owned by Sakichi Toyoda. As in textile industry there were many loom machines, off course! The function of loom machine is to make bundle of threads. Sakichi Toyoda invented a loom machine which would stop automatically if any of the thread would either break or gets cracked. This invention not only reduced waste but also assured quality. (Always fix the problem there and there, else it will make more trouble later.)
2. In 1930 Toyota Group set their automobile plant. It was headed by Kiichiro Toyoda, the son of Sakichi Toyoda. So as to have basics in automobile industry he visited USA to study Henry’s Ford Company. Here he found the concept of conveyer belts. The conveyer belts, we can see them on airport where luggage is moving on belts and passenger picks their respective luggage. He was impressed to implement the same but with additional automation. The part which was taken from conveyer belt was registered to make again. Means as the part is utilize the part was again assigned to manufacture. In this way Kiichiro Toyoda invented the term “Just In Time” concept. As soon as item is used it should be filled with new one. (As soon as a major module is completed it should be shown to product owner or stack holder.)
3. Further in 1940, Taiichi Ohho, made more automation in “Just In Time” production. He took each conveyer as customer of preceding conveyer, which made the current conveyer the supermarket to the following conveyer. This not only reduced the production time but overall efficiency of production also got increased. This build a “Pull System” rather than “Push System”. This became *Kanban System,* which provides for conveying information in and between the processes. (While a module is in progress there should be regular communication between team member, product owner and stack holder)
4. With above three system Toyota Motor Company formulated some basic principles they are:
5. Continuous Improvement. (Don’t fix any rule, just make it vibrant to make it simple and more useful, check out A, B and C point its continuous improvement.)
6. Respect for the people. (Make environment more joyful rather bossy.)
7. Long term philosophy. (Make risk handled first! Simple rule if risk is handled from start it gets reduced, rather keeping it to handle later will always increase the risk.)
8. Choosing right process to gain right results. (Iterate to get right process, hence direction will automatically open in right way.)
9. Imparting inductions to people and partners. (Don’t assume all people know everything!)
10. Continuous solving root problem, drives organization learning.

With above knowledge and principles in 1983 Toyota Motor Company, under name Lexus started to achieve the world best car and in 1989 Lexus did it! Now with this example if we replace *the car* with a product or I must say with a project, the same principles will drive our project with best results and an appropriate product for the stack holders. And the same methodology becomes Agile Methodology.

Confused? Means how the model of car can replace model of a software project? So let’s come out of the history and understand that they cannot be same, and hence we have Agile Software Development. Which is based on above principles but with some more additions for software development purpose.Let’s see what the Non-Agile and Agile Approaches are.

A Non-Agile Approach: (Waterfall Approach) A software development process has Application Life Cycle Management (ALM). This is a routine work most organization follows. The main concept is to gather requirements and document it down. This is done by business analysts or product manager. This is further taken by engineers and architecture which laydown the function requirement, by either prepare the flow chart or case diagram, which ever suitable and readable by technical team. Next development starts where developers write software code. Next, the testing phase, test cases are written to validate the specification and testing is carried out, if found defect the bug is assign back to development team, till the product is achieved. Well in my view doesn’t it more theoretical rather than a real world example. I am thinking so because by the time testing is complete many features may have been changed or may not require and the requirements would be differ as laid out in the beginning! Now chaos start between the entire team and the stack holder. Well it happens!

An Agile Approach: The same Application Life Cycle Management (ALM) is taken in Agile Methodology. But it is taken into small chunks. Requirement is gathered not only by business analysts or product manager, but also with development team and testing team. The required document is made which off course will be small and while making any technical or functional document, if there was gap in gathering requirement it is again discussed with stack holder. Parallel development on the clear requirement starts, test starts and if there was change in requirement that is coupled and testing is perform. If validation and verifications are done okay, this requirement is delivered. The next chunk or requirement is gathered and process is repeated again till the project is completed. So basically An Agile Software Development is continuous integration till the product or project is completed.

As we can see from above the latter being more practical and possible. Let’s go further in detail with Agile Methodology. As given above the small chunks of entire project is termed as “Scrum” in Agile Methodology. And the operative force is not “Push” it is “Pull” means it implements the *Kanban System* as describe in our *the best car* example.

Scrum: It is a real world progress of the project, it is not based on best guess, imagination or uninformed forecast, but is real plan and schedule release. Scrum can be a module, set of modules or a project. Each task in Scrum, can be termed as *Sprint* making it run over a week or two, when all sprints are complete the scrum is tested and delivered to stack holder, who looks the progress of the project. Hence Scrum allows project either to adjust or reoriented based on completed work. Here we can see there is no imaginations, speculations, assumptions, predictions or one man leader, it is simply real situation scenario. Let’s see what the roles of Scrum are:

1. Product Owner: This role is taken up by business analysts or product manager. This role is in between stack holder (client) and the team member. The role is responsible to understand the client interest and development team understanding. The responsibility of this role is to completely analysis the requirement and prioritize, which acts as starting the first Scrum.
2. Scrum Master: The role acts as a facilitator between Product Owner and the development team. Please note that Scrum Master is not a Project Manager or Team Leader and neither manages a team. Scrum Master Works to remove any bottle neck team is facing to achieve the Sprint goals. Also scrum master advice the product owner how to maximize the return of investment (ROI) of the team. This directly increases the creativity, productivity and quality of the development team.
3. Development Team/Team Member: For software development Agile Methodology recommends a team having software developers (programmers), UI designer(s), architect, analysts, QA experts and QC testers.

We can see that a project goes with real life sample providing Team Member autonomy, similar to the Product Owner, both have freedom accompanied with responsibility towards the project. The best part is there being no project manager, team leader or any one taking entire responsibility, but it’s a team driven where stack holder is also participating towards the project. ☺

Kanban: As we have seen in example of *the best car*, to understand Agile Methodology that Kanban is just in time production and the production of most demanded product, by conveying information in and between the processes. The most promising part is Kanban removes bottleneck if it occurs to achieve the product goals, the same is achieved in Agile Software Development by taking basic steps as follow:

1. While requirement is gathered, analysts, architect, programmers and testers are included. Each brains works in parallel to achieve the same goal.
2. Documentation is done as a user story, Agile says only document what is required in small span of time, by just considering, Who? What? Why? And When? Simple.
3. 15 minutes meeting in a day before and after work. Hence communicating more and documenting less. Documenting to make understand stack holder what is running and communication on what task to start, what is backlog before starting work, what is done and completed, what will be done tomorrow at the end of the day.
4. Creating prototypes first, in the form of sketches or a real working model in such a way that can be directly implemented into sprint.
5. Planning to test early and more frequently, which makes real sense that product does not have missed some important configurations.
6. Delivery more frequently, which makes product owner to take the stack holder in confidence, while the stack holder is aware of the entire progress of the project.

Above given are the more than sufficient examples and points to understand what Agile Software Development is. While if we combine the both CMMI Levels and Agile we can see that both co-exist easily. Means if we go agile way we have short documentations but at the end of Scrums we have complete documentation. The magic is the documentation and the project both complements each other. The same set of rules are listed in CMMI levels. Hence most CMMI appraised organization relay on Agile Methodology to achieve product success.

But both CMMI and Agile Methodology is not a key for project success, project may get fail. In this scenario all are responsible from team member, scrum master, product owner and stack holder. As in Agile Methodology there is more or less no communication gap, each user stories (tasks), sprints and scrums are documented, next scrum is taken up only with the success of the preceding scrum, and the end product resembles the document which is already communicated, so we can clearly see that there is no violation of any rules which is describe and set by CMMI.

With some clear light on CMMI and Agile Methodology for Software Development, we can assume now is the time to implement the same with a practical more realistic example. So let’s start with a sample project. Before starting, it will be good to go once again to the top of the article and read one more time, as I will try to put each and every accepts of CMMI and Agile Methodology in the sample project.

**ASP.NET sample project in Agile Way:**

*(Please Note: For Agile development there are many tools available to track communication, creating scrum, sprints, documentation, etc. But I will leave on you whether to select those tools. In either way for documenting communications Word Pad is enough, for creating scrums and sprints, tracking backlog, etc. excel is good. I will not use sketches rather dummy module will be used, such a way that it can be integrated in our project. Off course need not to say that Visual Studio is required with 4.0 minimum or above, my code language is C#, you can take your favorite .NET language. An intermediate or expert in ASP.NET programming is required, above average on generics, design patterns and logical thinking is expected, little know of JavaScript, HTML and CSS will be add on. Any other technology required related to the project, is the challenge and test towards Agile Methodology, to see how it is use to take care of these bottlenecks, that’s why a sample project is taken into consideration. )*

So as to implement a project, we will have to make assumptions of some common concepts. We are directly going on to create project which does not include initial acquisition and marketing of the client, neither after completing service model is taken into consideration. Rather a client is already acquired and a product owner or product manager with in an organization is assigned to the client.

Let’s assume that there is first time conversation going on between Stack Holder (The Client) and Product Owner. Stack holder gives the brief of the project and what he is looking as a solution.

What client is looking? The first communication between Client and Product Owner:

***Client****: We have many portal running and all portal are based on ASP.NET. As portals are running there are lot of configuration elements which portal contains. Hence the web.config file is completely messed up. What we are looking is the solution to separate the default configuration of ASP.NET and the portal configuration in separate configuration file. This should be such implemented that our configuration file should be read on application start and kept the elements in memory for entire application cycle. The configuration can be read at many moment by the developers and should be easily searchable. We have access on global file provided by the .net framework so reading and disposing once will be not be an issue. The solution we are looking should be generic, applicable to any configuration file (off course that will be xml based) and easy to implement and used in various portals.*

***Product Owner (Project Manager)****: Sure, means we need to just read the configuration file and keep the same in memory which can be searched any time. What I am getting from the requirement is the plugin you are looking which can be used in your any portal so that web.config remains simple and all your configuration details are moved into separate configuration file. As it is cleared that you have complete access over your portal and hosting environment, adding a dll would not be an issue for the configuration. We can provide you a plugin and easy one time step to implement the same. WE request you to provide the language you are using in .net and a sample configuration files over which we can test and provide you a mock running project.*

***Client****: Yes sure we will soon provide you configuration files, more or less we are using 4.5 framework of .net and as its been suggested that the plugin will in the form of dll, we suggest you choose the language of development, that ultimately will not be an obstacle for the plugin. But the implementation part should be generic and easy to configure without any further complication in we configuration file.*

***Product Owner****: Sure, the same will be uphold and a mock dummy project will be shown to you. Further we are waiting for the sample configuration files.*

As we can see that client needs a plugin that can read an xml file and the data once read should be available in searchable form and should be kept in memory. While there can be any valid xml file which needs to be read and kept in the memory. So let us now start the process of scrum and sprint and make the task list. The scrum document should be made in presence of architecture, development team leader and quality control team leader. This will be insure that what will be developed and what needs to be control for quality and quality testing.

Just download the folder “AgileProject” and unzip. In the same there are three folders:

1. Documentation: This folder will be used by all to maintain Issue Logs (Back Logs), Scrum and sprint (user stories) and burn down chart.

2. Project: This folder is actual working project for the requirement. Here the same had been maintain in days. Off course there will be versioning which should be maintained using VSS, Team Foundation Server, etc. any of your choice.

3. Sample XML: This is the folder having configuration files which client had provided the same will be used in our project.

A. Open “Documentation=>Day1\_7” folder and the “Project\_Tracking” excel file. There are three sheets, let’s go one by one and see what this offers.

Sheet 1 (Issue Logs): Column “User Story”, gives us a brief about what the task would be. Column “Task Description” provides us with the brief of the user story. Column “Task Holder” gives information who will be doing the task. Column “Estimation” provides information how much time will be required to complete the task, in days. Column “Priority” gives which task is to be done first and then after. (This is also called dependency). Column “Task Status” depicts status of task, and “Comments” column provides comments about the or on the task.

In the first sheet we define the task so that we can give a demo to the client. While on other side what impact or further task needs to be done or needs to confirm from the client is also entered. As of we only need to provide a demo which either client will just watch or even run the demo from his/her desk on our test hosting. Hence we would require that this demo should be bug free or have least error so we also included the QA and QC in the same demo. Here we can see that task index are not in sequence, it has a reason, as we discuss the requirement we come across different things which then are kept under priority.

Sheet 2 (Iteration 1): This sheet needs to be explain in depth. Understanding of the same is core concept to run or obey the Agile way working. There are two tables on left is same as of sheet one (Issue Log). We can notice that we have taken only those task id which are required for demo. Rest we will be keeping for next iteration or will be consider after client discussion. Below left side table is Total Days we have to make demo or release of version 1.0.0.1. (We have 6 days). Before going to “Calculated Available or Assign Time”, we need to see table on right side.

Table on right has some calculation over estimation, following is clarification about same.

|  |  |  |  |
| --- | --- | --- | --- |
| No. |  | Days | Explanation |
| 1 | Start Date | 3/2/2015 | Take your start date of the project. |
| 2 | No. Of Working Days | 6 | Time estimated for demo release in days. |
| 3 | No. Of Developer & QC | 3 | We have 2 developers, who also acts as QA & 1 QC for testing, hence team of 3 people |
| 4 | No. of days for development & QC | 18 | No. 2 \* No. 3 of Column “Days“ i.e. (6\*3) |
| 5 | Efficiency Factor | 0.69 | If we have 8 or 9 hours working in a day we can’t take 100% working hours. Ideally 0.7 is the standard which is taken. |
| 6 | Actual development & QC days | 12.42 | No. 4 \* No. 5 of Column “Days” i.e. 18\*0.69 |
| 7 | End Date | 3/8/2015 |  |
| 8 | Backlog as per efficiency. | -2.07 | This one is tricky. It assumes that on day one before starting actual work there is backlog, which further gets reduced. Its calculation is (Actual development & QC days)/(No. of working days) \* -1. We multiple by -1 to create backlog which needs to be cleared. |
| 9 | Days used for backlog | 3 | As we assume backlog of -2.07 the days utilize for the same is (No of days for development & QC)/(No. of days of working) |

Now let us define “Calculated Available or Assign Time” which is below left table. This is an average time available. This is calculated as “Actual development & QC days” (from right table) – “Total Days to release” (sum of estimated time from left table) i.e. 12.42 – 6 = 6.42.

Sheet 3 (Iteration 1 Burndown): Little bit more complicated table, but as we go defining each column it becomes clearer. In this sheet we enter actual data for each developer, QA and QC. So let’s see what it offers.

|  |  |
| --- | --- |
| Work Day (Col A) | It is actual working day in an organization. Off course holidays and week offs needs to be excluded. |
| Work Days (Col B) | Incremental number of days the project is under developing. This should always start from Zero. |
| Ideal Remaining Effort (Col C) | We have calculated backlog which is -2.07 and actual working days available which is 12.42. Hence remaining effort becomes (calculated backlog \* current working day count) + working days available. Which is (-2.07 \* 0) + 12.42 = 12.42, and (-2.07 \* 1) + 12.42 = 10.4, and so on for each incremental day. |
| Col D | We will define this at the end.\* |
| Col E and F | We have 2 developers. Each one enters there time consumption over each task as the day ends. It should be sum of their allocated task estimation (in days) which is sum of task ID 1, 2 and 3, in sheet “Iteration 1” left table. These task IDs are actual development work. |
| Col G and H | We have 2 QAs. Each one enters there time consumption over each task as the day ends. It should be sum of their allocated task estimation (in days) which is sum of task ID 7, 8 and 9, in sheet “iteration 1” left table. These task IDs are actual QA task cross checking each other’s work. |
| Col I and J | Here QC check the bugs if found raises it either simultaneously developer fix the bug or is taken separately. Task IDs are 11, 12 and 13. |
| Total Task Completed (Col L) | As the working day is completed, total complete task for the day is calculated. As obvious it is sum of Col E, F, G, H, I and J. |
| Actual Remaining Task (Col M) | This is calculated from total estimated days for completion – total completed estimation in a day. We have total days 6 and sum of completed in col L. We can see in row 2 the same 6 – 0 = 6 on day zero. |
| Man Day Used (Col N) | = Backlogs days (3) \* current incremental day. For row 2 we have : 3 \* 0 = 0. |
| Calculated Efficiency Factory Col O | = Total Task Completed / Man Day Used, we have formula in excel if Man Day Used is zero we take efficiency a zero or we use the formula. |

\*Col D is simple the copy of Col M, we can see from the heading. Col D is used just to include Col B, C and D in excel graph to create the “Burn Down Chart” Col D is marked in light pink to be ignored for understanding, but to include in graph.

From above “Project\_Tracking” document we have a rough ideal how to estimate a module (Scrum) and use for the benefit to control the project and define each expect of the project in “Issue Logs” Hence same is used each day. Each story becomes sprint, collection of sprint which can be delivered becomes Scrum, and continues integration goes on till we complete the requirement of the project. The “Burn down Chart” provides us whether we are on the time or lagging behind, if so necessary steps are taken to bring down the development on time.

*(The same excel style and calculation is used for each scrum, the same calculation is applied to complete the requirement of the project)*

B. Now Open “Project\_Day1\_7”. We have .NET website project. It contains default.aspx. You can just download it. Following are the functionality:

i. User can browse any XML file.

ii. A button is given to valid whether file is a valid XML file.

iii. Other button reads and displays the content of the file and keeps it in memory.

*The demo is shown to the client. The ability to keep data in memory, searching any attribute or xml element is the concern of the client now. Also as per initial discussion client needs an API so that above mock code can be run and implemented across any of his portal. Also in discussion with client, client asked to create separate log utility. The log utility configuration should be kept in main web configuration file.*

Now open folder “Documentation->Day9\_14”. We have again project\_tracking excel file with further more stories. Implementation of these user stories are carried out for developing code.

Open folder “Project->Day9\_14”. There are two projects, they are

1. ReadXML: which implements all the necessary codes for reading configuration files and keeping the content of configuration in hierarchy as well as in simple generic list so that required xml elements, keys and attributes can be searched. The website demonstrate the usage of some of the functionality of search.
2. ReadXML.Logger: Which implements the code for logging errors and the configuration needs to be set up in the web.config of website. (IF implemented in windows project the configuration is set up in app.config file)

Each story is implemented in above project while completion of each stories in the form of days is entered in iteration worksheet of “project\_tracking”.

The last part “Documentation->Day16 folder, I am just keeping it blank. One can fill in estimation and development/QA/QC. Just fill in the iteration sheet burn down chart will response as per completion of each day in the form of day. In Project->Day16, I am keeping complete code implemented for the plugin. Please review the web.config file for configuration. Plugin start set up in global.asax and implementation of usage in default.aspx.cs. You can just evaluate the process and fill the project\_tracking excel. Rest documentation I am not including keeping the tracking in development part itself.

**Points of Interest:**

Well, there are many points which conclude my article. Learning and living agile is great. Its entire team effort, while the best is pulled out in reasonable time. Not every point I had covered, but tried to put some introduction over CMMI levels and how CMMI upholds along with Agile methodology.