Agile Methodology for Complex Software Development in Large Scale Organization
(Complex Software Classification Based On Different Metrics and Managerial Tracking Of The Project)

Suhail Ahmed. A
Christ University, Bangalore-68

Abstract: Agile is a software development methodology. Agile development of software will be carried out on different sprint. In each sprint the developer will be given requirement to develop or enhance the application. While developing the small application or enhancement agile methodology work smoothly as the complexity of project increases. It tend to give more pressure to developer, client and financial and project management team. As managing the complex software becomes more difficult to project lead and managers because of complex project requirement will be more complex. Understanding the requirement will take more time. Execution of the requirement will be taken more time either will be completed in same sprint or further backlog will be added. Difficulty of developing the software arise now because to cover up existing back log and to develop new requirement within time specified with the existing resource, managing the user stories and managing the different development team which is geographically scattered in different places. As this agile methodology improvised by using the different agile tool and closely monitoring the daily sprint activities and more concentration should be given to project management by management team project management and operational managers. Agile model believe that every development activities need to be handled uniquely and the other methods need to be modified much to suit the project requirements. In agile the requirements are divided and conquer to small time frames to deliver specific features for a sprint release. Cyclic approach is carried out again and again in development of software.

Keywords: Agile, metrics, team, developer, calculation, software, testing, Programming, estimation, scrum, scrum master.

1. INTRODUCTION
The software market has grown exponentially over the last decade. Consequently the easy make has further matured and is constantly actively engaged in the software development. Further, complete changes in service rules are at the drop of a hat required forthcoming implemented in the software, herewith increasing the urge on developer in twist implementation big idea time. Traditional Agile software arts and science solutions are not considered helpful in the rich organizations. These methodologies tackle ideas on Team pregnancy and agility; all the same these are to a great degree designed for low and augur size projects. The adoption of nimble methods in a ample organization has regular proved challenging by profuse organizations and authors. There are absolutely few reports practically the born with a silver spoon implementation of Agile behavior in oracle sized corporations, ultimately with sprinkling modification.

The adoption of agile methods in a large organization has often proved challenging by many organizations. Most of the time project ends up with loss in profit or delaying the deliverables which leads to loss for the large organization using the agile development process.

Agile has been conceived to keep perpetual communication by the whole of the clients so that culmination in engagement in activity application requirements gave a pink slip be tackled no ifs ands or buts about it and in time.

Scrum is more love administration process alternative than a knowledge process. It is malleable enough subsequent combined by the whole of other all right already software habit processes. Shows that during scrum provide the material to accompany development made up for lost time, anyway it lacks structured approach.

The adoption of agile methods in a large organization has often proved challenging by many organizations. but this can be adopted to complex project by modifying some of the development activities.

2. Classify the undertaking based on accessible metrics

Different Types of Metrics calculation
1. Chidamber and Kemerer Metrics
Chidamber and Kemerer originally defined the CK metrics in 1991-1994 to find the complexity of coding of software

- **WMC Weighted Methods Per Class**
  
  WMC = number of methods defined in class

  Distinctive points of function or method have been characterized. One way to set upper limit the number of methods in a class to, say, 20 or 50. Another way is to specify that a maximum of 10% of classes can have more than 24 methods. This permits expansive classes however most classes ought to be little.

- **DIT Depth of Inheritance Tree**
  
  DIT = maximum inheritance path from the class to the root class

  The more inheritance in a class hierarchy, the more function and properties it is likely to extend, making it more complex. Inheritance is a tool to manage complexity, really, not to increase it. As a positive factor, inheritance promotes reuse because of functional inheritance.

  A recommended DIT is 5 or less. DIT <= 5 because excessively deep class hierarchies are complex to develop. Some sources allow up to 8.

  Special cases. When a class inherits directly from System. Object (or has no Inherits statement), DIT=1. For a class that inherits from an unknown (unanalyzed) class, DIT=2. This is because the unknown class eventually inherits from System.

  Object and 2 is the minimum inheritance depth. It could also be more

- **NOC Number of Children**
  
  NOC = number of immediate sub-classes of a class

  NOC equals the range of instantaneous sub training derived from a base magnificence. In conventional Visual Basic inheritance isn't available and for this reason NOC is usually 0.

  A high NOC, a massive number of infant instructions, can imply several matters:
  
  - High reuse of base class. Inheritance is a shape of reuse.
  - Base elegance may additionally require more trying out.
  - Improper abstraction of the parent elegance.
  - Misuse of sub-classing. In any such case, it is able to be important to institution related instructions and introduce every other stage of inheritance.

  High NOC has been observed to suggest fewer faults. This may be due to high reuse, which is perfect.

  A magnificence with a high NOC and a high WMC shows complexity on the pinnacle of the magnificence hierarchy. The elegance is probably influencing a large quantity of descendant instructions. This may be a sign of poor layout. A redecorate may be required.

  Not all classes need to have the equal range of sub-training. Classes higher up within the hierarchy need to have more sub-classes then those decrease down.

- **CBO Coupling between Object Classes**
  
  CBO = number of classes to which a class is coupled

  Two instructions are coupled whilst methods declared in a single class use strategies or instance variables described by means of the other class. The uses courting can move either way: each makes use of and used-via relationships are taken into consideration, however most effective as soon as.

- **RFC and RFC’ Response for a Class**
  
  The response set of a category is a fixed of techniques which can probably be carried out in response to a message acquired by using an item of that class. RFC is honestly the quantity of techniques inside the set. RFC = M + R (First-step degree) RFC’ = M + R’ (Full measure) M = variety of strategies within the magnificence R = number of faraway strategies without delay referred to as with the aid of methods of the magnificence R’ = range of faraway techniques known as, recursively through the entire call tree massive RFC has been determined to indicate extra faults. Classes with a high RFC are greater complicated and harder to apprehend. Testing and debugging is complicated. A worst case value for possible responses will help in appropriate allocation of trying out time.

- **LCOM1 Lack of Cohesion of Methods**
  
  The 6th metric in the Chidamber & Kemerer metrics suite is LCOM (or LOCOM), the lack of cohesion of methods. This metric has received a great deal of critique and several alternatives have been developed. In Project Metrics we call the original Chidamber & Kemerer metric LCOM1 to distinguish it from the alternatives.

<table>
<thead>
<tr>
<th>System</th>
<th>J2ee</th>
<th>J2ee</th>
<th>C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes</td>
<td>46</td>
<td>1000</td>
<td>1617</td>
</tr>
<tr>
<td>Lines</td>
<td>50,000</td>
<td>300,000</td>
<td>500,000</td>
</tr>
</tbody>
</table>
Quality    | L  | H   | M
---|---|---|---
CBO     | 2.48 | 1.25 | 2.09
LCOM1   | 447.65 | 78.34 | 113.94
RFC     | 80.39 | 43.84 | 28.6
NOC     | 0.07 | 0.35 | 0.39
DIT     | 0.37 | 0.97 | 1.02
WMC     | 45.7 | 11.1 | 23.97

Table 1. Lack of Cohesion of Methods

### 2. Mood Metrics

Progressively, question situated estimations are being utilized to assess and foresee the nature of programming. A developing collection of experimental outcomes underpins the hypothetical legitimacy of these measurements. The approval of these criteria requires displaying that the metric measures what it indicates standard measure (e.g., a coupling metric really measures coupling and the metric is connected with a basic external metric, for example, unwavering quality, viability and blame inclination. For example, unwavering quality, viability and blame inclination. Regularly these measurements have been utilized as an early predictor of these remotely noticeable qualities, in light of the fact that the remotely obvious characteristics couldn't be measures until past the point of no return in the product advancement handle.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Acceptable Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling Factor</td>
<td>L</td>
</tr>
<tr>
<td>Lack of Cohesion of Methods</td>
<td>L</td>
</tr>
<tr>
<td>Cyclomatic Complexity</td>
<td>L</td>
</tr>
<tr>
<td>Attribute Hiding Factor</td>
<td>H</td>
</tr>
<tr>
<td>Method Hiding Factor</td>
<td>H</td>
</tr>
<tr>
<td>Depth of Inheritance Tree</td>
<td>L</td>
</tr>
<tr>
<td>Number of Children</td>
<td>L</td>
</tr>
<tr>
<td>Weighted Methods Per Class</td>
<td>L</td>
</tr>
<tr>
<td>Number of Classes</td>
<td>H</td>
</tr>
<tr>
<td>Lines of Code</td>
<td>L</td>
</tr>
</tbody>
</table>

Table 2. Mood Metrixs

**MOOD IMPLEMENTATION METRIC**

2.1 **Coupling** the Coupling Factor (CF) is surveyed as a segment. The numerator addresses the amount of non-legacy couplings. The denominator is the most extraordinary number of couplings in a structure. The most extraordinary number of couplings fuses both legacy and non-legacy related coupling. Legacy based couplings rise as decided classes (subclasses) secure systems and qualities shape its base class (superclass). The CF metric is joined into the MOOD metric suite.

2.2 **Cohesion**

Union insinuates how eagerly the operations in a class are related to each other. Union of a class is how much the close-by systems are related to the area illustration calculates the class. CK estimations suite examines the Lack of Cohesion (LOCOM),

2.3 **Encapsulation**

Encapsulation in Java is an instrument of wrapping the data (variables) and code following up on the data (techniques) together as a singular unit. In epitome, the components of a class will be maintained a strategic distance from various classes, and can be gotten to right through the strategies for their present class. Thus, it is generally rung data covering.

2.4 **Attribute Hiding Factor (AHF)**

The Attribute Hiding Factor measures the intangibilities of properties in classes. The imperceptibility of a characteristic is the rate of the aggregate classes from which the quality is not obvious. A property is called obvious in the event that it can be gotten to by another class or question. Properties ought to be “covered up” inside a class. They can be kept from being gotten to by different protests by being pronounced a private.

2.5 **Method Hiding Factor (MHF)**

The Method Hiding Factor measures the imperceptibility’s of techniques in classes. The imperceptibility of a strategy is the rate of the aggregate classes from which the technique is not unmistakable.

The Method Hiding Factor is a portion where the numerator is the aggregate of the imperceptibility’s of all strategies characterized in all classes. The denominator is the aggregate number of techniques characterized in the venture.

2.6 **Inheritance**

Where one class procures the properties (techniques and fields) of another. With the utilization of legacy class the data is made reasonable in a various leveled arrange.

The class which acquires the properties of other is known as subclass (inferred class, tyke class) and the class whose properties are acquired is known as superclass (base class, parent class).

2.6 **Depth of Inheritance Tree (DIT)**

The profundity of a class inside the legacy chain of importance is characterized as the most extreme length from the class hub to the root/parent of the class pecking order tree and is measured by the
quantity of precursor classes. In cases including numerous legacy, the DIT is the greatest length from the hub to the base of the tree.

Very much organized OO frameworks have woods of classes as opposed to one vast legacy cross section. The more profound the class is inside the chain of command, the more noteworthy the quantity of strategies it is probably going to acquire, making it more perplexing to foresee its conduct and, accordingly, more blamed inclined. More profound trees require more noteworthy plan many-sided quality, since more strategies and classes are included. To be sure, profound pecking orders are additionally an applied uprightness concern since it gets to be distinctly hard to figure out which class to practice from. Also, interface changes inside the tree must be reflected all through the whole class tree and question cases. In any case, the more profound a specific tree is in a class, the more noteworthy potential reuse of acquired strategies.

Applications can be thought to be "beat overwhelming" if there are excessively numerous classes close to the root, and sign that planners may not exploit reuse of strategies through legacy. On the other hand, applications can be thought to be "base substantial" whereby an excessive number of classes are close to the base of the progression, bringing about concerns identified with plan unpredictability and reasonable respectability.

3. QMOOD

The QMOOD measurements are recognized so measurements processed on a framework can be utilized to make sense of a sort of super metric, the Total Quality Index outline (TQI). Like the DISPOSITION measurements, the QMOOD measurements are characterized to be calculable ahead of schedule in the outline procedure. Bansiya and Davis initially settled on a gathering of design quality qualities construct freely in light of the qualities characterized in the International Standards Organization (ISO) 9126 standard: reusability, flexibility understandability, usefulness, extendibility, and viability.

4. Techniques to be adopt to complete complex software project by Project managers

Sprint Burndown Scrum bunches deal with headway into time-boxed sprints. Toward the start of the sprint, the gathering figures how much work they can complete in the midst of a sprint. A sprint burndown report then tracks the finish of work all through the sprint. The x-center addresses time, and the y-center point insinuates the measure of work left to complete, measured in either story centers or hours. The goal is to have all the assessed work completed before the complete of the sprint.

A gathering that dependably meets its figure is a persuading advertisement for composed in their affiliation. In any case, don't allow that to tempt you to fudge the numbers by declaring a thing complete before it genuinely is. It may look awesome incidentally, yet as time goes on just hampers learning and change.

**Epic and Release Burndown**

Epic and release (or frame) burndown charts track the progress of change over a greater gathering of work than the sprint burndown, and coordinate headway for both scrum and kanban bunches. Since a sprint (for scrum gatherings) may contain work from a couple of adventures and variations, it's key to track both the progress of individual sprints and furthermore legends and structures. "Scope creep" is the imbuenment of more necessities into an earlier portrayed augment. For example, if the gathering is passing on another site for the association, scope creep would ask for new segments after the hidden necessities had been delineated out. While continuing expansion creep in the midst of a sprint is appalling practice, scope change inside legends and variations is a trademark result of apt progression. As the gathering goes through the wander, the thing proprietor may go up against or empty work in perspective of what they understand. The epic and release duplicate down charts keep everyone aware of the forward and backward development of work inside the epic and variation.

**Velocity**

Velocity is the basic amount of work a scrum gathers finishes over the span of a sprint, measured in either story components or hours, and could be extremely helpful for calculation. The item proprietor can utilize pace to are expecting how rapidly a group can canvases through the accumulation, because of the reality the archive tracks the determined and finished work over various iterations—the more emphases, the additional right the conjecture.

suppose the item proprietor yearnings to whole 500 story figures inside the overabundance. We realize that the change group more often than not finishes 50 story variables for every new discharge. The item proprietor can sensibly suspect the group will require 10 cycles (give or take) to finish the predefined work. it's essential to screen how speed advances after some time. New gatherings can hope to peer a blast in speed in light of the fact that the gathering upgrades connections and the works of art technique. current groups can music their speed to guarantee enduring execution after some time, and can affirm...
that a chosen procedure trade made enhancements or now not. A lower in like manner speed is generally a flag that a couple some portion of the gathering’s change way has developed to be wasteful and should be included at the resulting review.

Control Chart

Control charts bring to a focus on the bi bike predate of abandoned issues—the total foreshadow from “in progress” to “done”. Teams mutually shorter bi bike times are reasonably to have higher throughput, and teams with continuous cycle times having a chip on one shoulder to copious issues are greater predictable in delivering work.

Measuring cycle foreshadow is an efficient and rolling with the punches way to regenerate a team’s processes for the results of changes are discernable almost willingly, allowing them to draw any further adjustments dose of one own medicine away. The conclude goal is to have a perpetual and short cycle has a champion start, trailing the entire letter of recommendation of work.

4. Techniques to be adopt to complete complex software project by development team.

Pair Programming to increased efficiency Agile Development in complex project

Pair programming is an agile software development technique in which two programmers work together at one computer. One programmer writes code while the programmer reviews each line of code as it is typed in. The two programmers switch roles frequently.

Pair Testing Pair mostly working is a software knowledge technique everywhere two twosome members function together at such keyboard to show the software application. One does the suspect and the at variance analyzes or reviews the testing.

Test-driven habit is a lifestyle technique to what place you must alternately set up a explain that fails already you write new sensible code. TDD is for abruptly adopted by hot software developers for society of research source conscience and is even being adopted by Agile DBAs for database development.

No contradict conscience reviews required

Better code quality

Effective communication

Higher team poise Better coding hast a weakness for adherence

Less belief on contrasting individuals

More know backwards and forwards inclusion of polished team members

Greater habit sharing in members

5. Implementation

Metrics can be found through the eclipse IDE and installing the required plugin to find the metrics. Different plugins are available to find the metrics in eclipse IDE.

Steps involved in adding the metrics

Installation procedure

Run Eclipse, go to Help menu -> Software Updates -> Find and Install

<table>
<thead>
<tr>
<th>Tabular Description of metrics values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safe Range</td>
</tr>
<tr>
<td>Number Of Parameter</td>
</tr>
<tr>
<td>McCabe Cyclomatic Complexity</td>
</tr>
<tr>
<td>Nested Block Path</td>
</tr>
<tr>
<td>Cohesion</td>
</tr>
<tr>
<td>Chidembar and kermer metrics</td>
</tr>
<tr>
<td>Hinderson and seller</td>
</tr>
<tr>
<td>Total Correlation</td>
</tr>
<tr>
<td>Pairwise Field Irrelation</td>
</tr>
<tr>
<td>Complexity</td>
</tr>
<tr>
<td>Cyclomatic Complexity</td>
</tr>
<tr>
<td>Weighted method per class</td>
</tr>
<tr>
<td>Lines of code</td>
</tr>
<tr>
<td>Lines of code</td>
</tr>
<tr>
<td>Other</td>
</tr>
<tr>
<td>Efferent coupling</td>
</tr>
<tr>
<td>Number of fields</td>
</tr>
<tr>
<td>Number of levels</td>
</tr>
<tr>
<td>Number of local in scope</td>
</tr>
<tr>
<td>Number of statements</td>
</tr>
<tr>
<td>Feature Envy</td>
</tr>
<tr>
<td>Feature Envy</td>
</tr>
</tbody>
</table>

Table 3.Metric Description
This tables describe the basic metric details for the project can be generated which can be used further for possible decision making and tracking of the project performance. The above criteria is basic parameter used by eclipse for generation of required reports to find the complexity of the projects.

6. FIGURES/CAPTIONS

SPRINT MEETING

Sprint meeting is carried out by the customer and scrum master for the purpose of allocating the customer requirement to developer and tester to complete the development task within the time constraint for delivering the product or deliverables.

Requirements Allocation.

Requirements are allocated to the different developer and team to achieve the deliverables in feasible way by the scrum master in scrum meeting.

Development and Test Deliverables

Development and testing of the assigned task is carried out by the developer and tester. Once the successful testing of task is carried out further process will be carried out.

Delivering the deliverables

Completed task are delivered to the customer as per requirement is defined. After successful completion of User Acceptance test the code will be integrated with the production code.

Changes in deliverables

The changes in the deliverables should be incorporated in the coming sprint cycle. Client is not satisfied with outcome of the requirements. Changes should be carried out next iterative sprint.

Backlogs of Sprint

Backlogs of the previous should be carried out in the next available sprint. Backlogs are generally carried out due to lack of clarification in the requirements or due technical challenges faced by the developer and tester.

Iterative Process of sprint.

The above process followed iteratively to achieve the requirement goals set out by the customer.

2. SECTIONS

Prioritizing Complex projects

1. Backlogs monitoring
2. Release deliverables plan
3. Resource Availability Calculation
4. Backlogs Review for every sprint
5. Sprint Planning
6. High level component sizing
7. Identifying dependent components.
8. Identifying Critical components.

Metrics For Complex Projects

1. Number of team involved in development of software
2. Geographical location of team.
3. Complexity of requirement and storage of data
4. Generating of reports

Different freeware Agile tools

- Easybacklog.
- IceScrum
- Agilefant
- SeeNowDo
- Agilo
- Xplanner
- XpWEB

Different license Agile tools

- JIRA tool
- Axosoft ontime scrum
- Lean Kit
- Microsoft Visual studio
- Telerik team pulse
- Rally Platform for Agile life cycle
- Plan Box Management

7. Agile Working

SCRUM

- Scrum is a dexterous - lightweight administration system that oversees programming advancement exercises utilizing incremental and iterative practices. Scrum is polished in associations to react changing prerequisites of the customers. Scrum concentrates on "What should be possible" rather than "Why it isn't possible". Consistent criticism of customers helps association to control and relieve hazard at early stages.
The advancement procedure is sorted out into sprints. Every sprint is goes on for around 30 days. Toward the begin of every sprint procedure holds a sprint meeting in which sprint length is chosen and the improvement group chooses the high need highlights from item build-up. Those picked components are further separated into errands to frame a sprint build-up. A day by day sprint meeting of 15 minutes is held toward the begin of the day. Amid this meeting every individual from the advancement group needs to answer the three inquiries. I.e. what was accomplished since the last meeting? What are you doing today? Furthermore, what will you accomplish before next meeting? The scrum ace is an interface amongst administration and scrum group and is in charge of evacuating obstructions to encourage the scrum colleagues. At the edge of the sprint, there is a sprint survey meeting in which advancement group shows possibly shippable item to get input. Toward the end, the advancement group holds review meeting in which handle change is altogether talked about.

**Actors in Agile**

1. Scrum master
2. Sprint meeting
3. Development Team
4. Testing team.

**Agile Working**

Agile Software Development values "working software everywhere comprehensive documentation". This core price tag asks us to visualize about how for all practical purposes and which kinds of documents are inadequate and when they prefer to be written. Agile Manifesto prefers “Working software everywhere comprehensive documentation”

Main purpose of agile is concentrate much on development and design of the software rather then document. But If organization wants to maintain both the development progress and planning information of development and well documentation can be maintained for the purpose of organization.

1. Training the resource in certification like ISO,CMMI
2. Emphasizing more on documentation activities and work

**8. Conclusion**

Above research on agile development concentrate on the classifying the project based on different metrics available namely CK metrics, Mood metrics, Q mood metrics to identify the project complexity and taking the sufficient measures to improve development and completing task with in time constraint.

Improvising of complex project can be done by following the pair programming technique, pair testing, and test driven development by development team.

Tracking of project can analyze by implementing report generation. Reports should be generated by the project manager to track the progress in development Sprint Burndown, Epic and Release chart, Burndown chart, Velocity chart, control charts.

**9. REFERENCES**


