Scrum Implementation in CBMS Project at Future Focus Infotech

Prepared by: Hari Thapliyal, PMP, MBA

Completed on: 02-Jul-12

Reviewed By: Seshan TR

Review Date: 06-Jul-12

Table of Contents

1. Problem statement 4
2. Objective 4
3. High Level Deliverables 4
4. Stakeholders Involved in Discussions & Reviews 4
5. High level suggestion for resources & projects 5
6. Responsibilities 5
7. Best Practices Adoption 6
   - 7.1. Product Backlog- Development/ Support 6
   - 7.2. Release backlog - Development 6
   - 7.3. Sprint backlog- Development 7
   - 7.4. Release Planning- Development 7
   - 7.5. Sprint Planning- Development 7
   - 7.6. Sprint Planning- FFI 9
   - 7.7. Process for picking job 9
   - 7.8. Planning Poker 9
   - 7.9. Scrum Meeting/ Daily Stand-up Meeting 11
   - 7.10. Regular stand-up meeting - FFI 11
   - 7.11. Weekly Sprint Review Meeting 12
   - 7.12. Sprint Review Meeting 12
   - 7.13. Sprint Retrospective Meetings 13
   - 7.14. Peer review 13
   - 7.15. Change Request Process 14
8. Velocity 14

9. Reports 14
10. Process of Doneness 14
11. Trainings 15
12. Guidelines 15
13. Notes 16
14. Proposed Development Workflow 16
15. Existing Development Workflow 17

1. Problem statement

- Agile/Scrum implementation in distributed team.
- Agile/Scrum implementation when the same team is working on both support and product enhancement projects
- Provide visibility to the customer in the form of templates provided by them
- Team is over working everyday but neither customer is happy about the outcome nor is the team happy because of burn out.

2. Objective

Implement an agile development methodology which produces expected results at the end of each sprint. Bring transparency between stakeholders and let them understand the related processes and their values.

3. High Level Deliverables

- Log maintenance - product backlog, release backlog, sprint backlog - define process, educate people, refine the process and enforce process.
- Project actual capturing. Define what type of actual should be captured at what stage. Define process, system and enforce the systems and processes.
- Define work sizing technique, baseline productivity/ velocity. Train people on work sizing methods and enforce process.
- Educate stakeholders about sprint and agile practices. Values, constraints and flexibility of this methodology.
- Define system for job pickup from sprint backlog
- Process and system for estimating and including Project management activities (Risk management, communication, stakeholder management, procurement management, change management, configuration management, team management) and support activities (system configuration, platform movement etc) in agile scrum cycle.
- Understand reporting requirements and define process and system for generating required reports in real time. Train the team, enforce periodic reporting and ensure the reported data is useful in decision making.

4. Stakeholders Involved in Discussions & Reviews

- One on one
    - Mr. Bharat Saoji
    - Mr. Seshan TR
    - Mr. Senthil
    - Mr. Padmanabhan (Pad)
    - Mr. Veeren
    - Mr. Nathan
- Team Discussion
    - Mr. Ramesh
    - Mr. Pawan
    - Mr. Elayaraja
    - Mr. Allwin
    - Mr. Radha
    - Mr. Karthikeyan
    - Mr. Thilagar

- Mr. Ashwin
- Mr. Sugumar
- Mr. Chandrasekhar

5. High level suggestion for resources & projects

Some of these assignments are already in place but message need to be communicated clearly with attached responsibilities. Other assignments are part of this consulting exercise.

- Ralph is Product Owner (PO)
- Darrell is Proxy Product Owner (PPO).
- Seshan is Scrum Master (SM).
- Identify three team and their members clearly to carry out support and development work smoothly.
    - Development Team Members:
        - Onsite Team Member: Darrell
        - Offshore Team Members: Allwin, Pawan, Elayaraja, Senthil
    - Support Team Members
        - Onsite Team Member: Pad
        - Offshore Team Members: Radha, Karthikeyan, Thilagar
    - QA Team Members
        - Sugumar, Ashwin

Note: This is high level suggestion and members can be swapped between dev and support team from one release to another and this depends upon their competency. Senior developers/ Old team members should not remain tied to support activities but they need to build competency of junior or new developers so that they can focus more on development work. Objective of the above team is to improve competency of team members and ensure support work, which needs much experience of the system and business, is not delivered by new or junior developer.

6. Responsibilities

- **Product Owner (PO) :** Product Demo, ultimate decision maker on requirement inclusion/exclusion from product backlog, final decision maker for requirement clarification. (Ralph)
- **Proxy Product Owner- (PPO)**: For development work he is a single point of contact to make decision or communicate decisions, requirement prioritization, explaining business requirements, review user stories and test case and remain available as and when needed by scrum team. There should be some overlapping of working hours between PO/PPO and scrum team. Prioritization can be done in rank field of Squish. Ensure information on Squish is latest always so that all team members and SM should refer to Squish system only. Without exception he shall ensure that requirement prioritization, ranking, clarification happen in the Squish system only. No emails for this work! (Darrell)
- **Scrum Master- (SM):** Remove blocks, maintain necessary documentation, develop processes and ensure people following scrum processes, guard the scrum team and

sprint goal so that the team can achieve the goal set for the sprint. Ensure that the scrum team does not work more than allowed time on support work.

- **Scrum Team- (ST):** Write user stories, test cases, estimate stories of the work in sprint backlog, enter actual efforts and status of work taken in the defined systems, report status in daily standup meetings and ensure workable product is shipped to customer. Team includes developers, testers, architect etc. ST should participate in Product Demo.

7. Best Practices Adoption

7.1. Product Backlog- Development/ Support

The product backlog is a prioritized features list, containing short descriptions of all functionality desired in the product. When using Scrum, it is not necessary to start a project with a lengthy, upfront effort to document all requirements.

A typical product backlog comprises the following different types of items:

- Features (product functionality)
- Bugs reported from previous sprint or current sprint
- Technical work (like feasibility, R&D, data migration, security etc)
- Knowledge acquisition (knowledge transfer, necessary documentation etc)

Support work / ticket is any work item which is not part of planned sprint.

While development items keep coming into Squish product backlog during the sprint is in progress, PO or his representative keep assigning release number to these items and alter release number of existing not completed work items. No work items whether it is support or development should be done without entering into the product backlog. Each ticket should have one assigned release number. Rank filed in Squish system should be used to prioritize the development work. In any given release two tickets should not have same rank.

Recommendations

- No requirement should be email to dev team, even no support ticket requirement on email.
- PO/PPO should make sure that development ticket has uniquely ranked items.

7.2. Release backlog - Development

Every release must have a release goal. Based on the released goal items should be assigned to any release number. Release backlog shall be finalized before planning sprint.

- Maintaining release backlog is PPO / PO's responsibility
- For the practical reasons release backlog is also maintained into the Squish system.
- Release backlog items are those which has release number assigned for next release and have been uniquely ranked
- Each release is typically comprised of 4 sprints. Based on the release items picked, number of sprints in any release can be changed.
- Items need to be delivered in any release shall be pulled into release backlog from product backlog and not from any other place

- Before committing items into a release backlog development team should conduct one release planning session to know the overall complexity of work involved in terms of impact on existing systems.

7.3. Sprint backlog- Development

Before start of any sprint, goal of that sprint shall be finalized. Items for any sprint shall be taken from release backlog only. If item is not in release backlog then that should not be reflected into sprint backlog.

- Each sprint is comprised of 2 calendar weeks
- For the practical reasons sprint backlog is also maintained into the Squish system.
- Priority can be defined in rank column of Squish system
- Sprint backlog should be developed while sprint planning at the start of every sprint
- PO/PPO should set the priority of work into sprint backlog
- Rank should have unique number
- Due to business need pending work items in product backlog can update, delete and add any time. Any change request in sprint backlog should follow the change request process.

Recommendation

- Only scrum team member should allocate work to each other/ self assignment
- Only scrum team member should estimate effort and time required and make commitment

7.4. Release Planning- Development

Every release is typically comprised of 4 sprints. PO, PPO, SM, ST should participate in release planning. Before start of first sprint of any release scrum team shall spend 10-12% of the complete release's planned development efforts in discussing and evaluating complexity of the squish development tickets. For example if one development release has allocated 800 hours development work then around 80 hours shall be spent in release planning. This is required because CBMS project is about enhance of existing system.

There are two defined artefacts that result from a release planning meeting:

- Release goal
- Release backlog with high level user stories and technical WBS

These items should be detailed enough so that development team can do the self assignment. This WBS should be developed in terms of impacted items and work need to be done.

7.5. Sprint Planning- Development

The Sprint Planning Meeting should be attended by the PO/PPO, Scrum Master, and the entire Scrum Team.

During the sprint planning meeting the product owner describes the highest priority features for coming sprint to the team. These items are picked from release backlog. The team asks enough questions that they can turn a high-level user story of the product backlog into the more detailed tasks of the sprint backlog. This step is about further refining user stories or WBS which was developed in release planning.

There are two defined artefacts that result from a sprint planning meeting:

- A sprint goal
- A sprint backlog

A sprint goal is a short, one- or two-sentence, description of what the team plans to achieve during the sprint. It is written collaboratively by the team and the product owner.

The following are typical sprint goals on an eCommerce application:

- Implement basic shopping cart functionality including add, remove, and update quantities.
- The checkout process—pay for an order, pick shipping, order gift wrapping, etc.

The sprint goal can be used for quick reporting to those outside the sprint. There are always stakeholders who want to know what the team is working on, but who do not need to hear about each product backlog item (user story) in detail. The success of the sprint will later be assessed during the Sprint Review Meeting against the sprint goal, rather than against each specific item selected from the product backlog.

The sprint backlog is the other output of sprint planning. A sprint backlog is a list of the release backlog items the team commits to delivering plus the list of tasks necessary to delivering those product backlog items. Each task on the sprint backlog is also usually estimated. Planning poker can be used for estimation.

An important point to reiterate here is that it is the team that selects how much work they can do in the coming sprint. The product owner does not get to say, "We have four sprints left so you need to do one-fourth of everything I need." We can hope the team does that much (or more) but it is up to the team to determine how much they can do in the sprint.

- Use the concept of epic for big feature to be implemented
- Use the concept of sub-stories for complex stories
- Use the concept of technical spikes for unknown technical complexities. Allocate hours for technical spike so that story writing and estimation can be done from the output of technical spike work. Efforts for technical spikes should be track separately.

- Use the concept of story telling and then story writing to know the detail of work involved in delivering a feature
- Develop WBS for technical work

This meeting should be conducted on Monday morning after the retrospective meeting. Make sure all required stakeholders participate in this meeting.

7.6. Sprint Planning- FFI

- Team members should have a prioritized items into release backlog with them before they start this process
- **Developing User Stories**
    - Team members should detail user stories while sprint planning. That may be high level items and can be refined while actual development.
    - Refined user stories should be reviewed and approved by PO/PPO/Originator.
    - User stories should be self assignable.
- **Planning Poker:** Use planning poker for estimating the story points sizing of the sprint backlog items.
- **Sprint Backlog:** After story sizing team should decide how many work items they can commit based on the story points. This commitment depends on team velocity and not on any individual.

7.7. Process for picking job

- Agile team should be a self disciplined team and they should pick the job without intervention based on their competency and ensure that all sprint items are picked up.
- If required they can help each other.
- It is not ticket based commitment that how many tickets individual are doing. No competition between team members.
-

7.8. Planning Poker

- Guidelines
    - Participants in Planning Poker include all of the developers on the team. Remember that developers refer to all programmers, testers, database engineers, analysts, user interaction designers, and so on.
    - At the start of Planning Poker, each estimator is given a deck of cards. Each card has written on it one of the valid estimates, card reads like 0, 1, 2, 3, 5, 8, 14, 20, 40, and 100. The cards should be prepared prior to the Planning Poker meeting, and the numbers should be large enough to see across a table.
    - Complexity definition
        - 0- No work involved
        - 1,2,3- Very Simple Complexity
        - 5,8 – Simple Complexity
        - 14- Middle Complexity
        - 20- Complex
        - 40- Very Complex

        - 100- Unknown Complexity
- Steps
    - Moderator, usually the product owner or an analyst, read items from release backlog
    - Team discusses the items and comes up with user stories for each item and put those items in sprint backlog
    - Each user story or theme to be estimated, a moderator reads the description.
    - To estimate the complexity of each story, estimators discuss the story and the product owner answers any questions that the estimators have. However, everyone should remember that at some point additional discussion does not lead to improved accuracy. The goal in planning poker estimating is not to derive an estimate that will withstand all future scrutiny. The goal is to develop estimate that can be arrived at cheaply.
    - After all questions are answered, each estimator privately selects a card representing his or her estimate. Cards are not shown until each estimator has made a selection. At that time, all cards are simultaneously turned over and shown so that all participants can see each estimate.
- Guidelines
    - It is very likely at this point that the estimates will differ significantly. This is actually good news. If estimates differ, the high and low estimators explain their estimates. It is important that this does not come across as attacking those estimators. Instead, you want to learn what they were thinking about.
- Steps
    - The group can discuss the story and their estimates for a few more minutes. The moderator can take any notes he thinks will be helpful when this story is being programmed and tested.
    - After the discussion, each estimator re-estimates by selecting a card. Cards are once again kept private until everyone has estimated, at which point they are turned over at the same time.
- Guidelines
    - In many cases, the estimates will already converge by the second round. But if they have not, continue to repeat the process. The goal is for the estimators to converge on a single estimate that can be used for the story. It rarely takes more than three rounds, but continue the process as long as estimates are moving closer together. It isn't necessary that everyone in the room turns over a card with exactly the same estimate written down. If I'm moderating an estimation meeting, and on the second round four estimators tell me 5, 5, 5, and 3, I will ask the low estimator if she is OK with an estimate of 5. Again, the point is not absolute precision but reasonableness. Do the Right Amount of Discussion
    - Some amount of preliminary design discussion is necessary and appropriate when estimating. However, spending too much time on design discussions is often wasted effort. Here's an effective way to encourage some amount of discussion but make sure that it doesn't go on too long.

- Place a two-minute timer tool in the middle of the table where planning poker is being played. If agreement isn't reached, the discussion can continue. But someone can immediately turn the timer over, again limiting the discussion to two minutes. The timer rarely needs to be turned over more than twice. Over time this helps teams learn to estimate more rapidly.

7.9. Scrum Meeting/ Daily Stand-up Meeting

- Remember the "Ham and Eggs" story, only committed people should speak in stand-up meetings.
- It helps setting the context of day's work.
- All team members are required to attend the meeting.
- This is information dissemination meeting therefore anybody who is interested in project status should join there as an observer.
- The daily scrum is not used as a problem-solving or issue resolution meeting. Issues that are raised are taken offline and usually dealt with by the relevant sub-group immediately after the meeting.
- Three questions of stand-up meeting are
    - What did you do yesterday?
    - What will you do today?
    - Are there any impediments in your way?

  Typical examples of impediments are
    - My ABCD broke and I need a new one today.
    - I still haven't got the software I ordered a month ago.
    - I need help debugging a problem with Rakesh.
    - I'm struggling to learn XYZ and would like to pair with someone on it.
    - I can't get the vendor's tech support group to call me back.
    - Our new contractor can't start because no one is here to sign her contract.
    - I can't get the ABC group to give me any time and I need to meet with them.
    - The department VP has asked me to work on something else "for a day or two."
- The daily Scrum meeting is not a status update meeting in which a boss is collecting information about who is behind schedule. Rather, it is a meeting in which team members make commitments to each other. If a programmer stands up and says "Today I will finish the data storage module" everyone knows that in tomorrow's meeting he will say whether or not he did finish. This has the wonderful effect of helping a team realize the significance of these commitments and that their commitments are to each other, not to some far-off customer or salesman.
- In cases where the scrum master cannot remove these impediments directly himself (like technical issues) he still takes responsibility for making sure someone on the team does quickly resolve the issue.

7.10. Regular stand-up meeting - FFI

- Every day stand-up meeting on phone at 8PM (IST)
- People have to give status update on 4 questions: a- what have they done, b- what are they planned for today/ (tomorrow for offshore team), c- any blocker, d- any help needed.

- Every team member must attend the call in case of telephonic stand-up meeting or must be in team area when stand-up meeting is happening.
- Due to any unavoidable situation if any skip is happening then update must be given to the peer team member before stand-up meeting starts and peer should update that status in stand-up meeting.
- This is one important ritual from team bonding perspective also therefore even if everybody is convinced that there is no update today and no stand-up meeting needed today, I suggest meeting each other saying hello and bye.
- Requirement change process. In between the sprint if any new work comes up or existing requirement changes then follow the requirement change process.

7.11. Weekly Sprint Review Meeting

Because of availability of PO he has assigned PPO on this project. PO reviews the product on weekly basis. On the weekdays if any scrum team has question then that is replied through PPO.

In the weekly demo process PO looks into the progress of running sprint items. During review a new requirement can come up, existing requirement can go out, existing / new requirement can take high priority so that must be done in the current sprint. Sometimes priority is set in such a way that completed work item cannot wait for final product demo of the sprint and that should be moved into production immediately.

Recommendation:
As per the customer's convenience this meeting is generally done on Friday night. It is suggested to create some overlapping time so that all offshore sprint team members can also participate in this process.

7.12. Sprint Review Meeting

At the end of each sprint a sprint review meeting shall be held. During this meeting the Scrum Team shows what they accomplished during the sprint. Typically this takes the form of a demo of the new features.

The sprint review meeting is intentionally kept very informal, typically with rules forbidding the use of PowerPoint slides and allowing no more than two hours of preparation time for the meeting. A sprint review meeting should not become a distraction or significant detour for the team; rather, it should be a natural result of the sprint.

Participants in the sprint review typically include the PO, PPO, Scrum Team, Scrum Master, management, customers, and developers from other related projects.

During the sprint review the project is assessed against the sprint goal determined during the Sprint Planning. Ideally the team has completed each product backlog item brought into the sprint, but it is more important that they achieve the overall goal of the sprint. Any spill over from the current sprint is moved into next sprint. If the spill over is happening from the last sprint of the release then one more sprint can be added in the release so that release goal is met.

It is also possible that there is nothing to show to the customer because important items has already been shown to him while weekly review or item has been delivered into the production.

In that case this ritual should be formally followed and a document of release items (delivered to UAT) of the sprint should be maintained.

Recommendation:
As per the customer's convenience this meeting is generally this meeting is done on Friday night. It is suggested to create some overlapping time so that offshore team can also participate in this process.

7.13. Sprint Retrospective Meetings

No matter how good a Scrum team is, there is always opportunity to improve. Although a good Scrum team will be constantly looking for improvement opportunities, the team should set aside a brief, dedicated period at the end of each sprint to deliberately reflect no how they are doing and to find ways to improve. This occurs during the sprint retrospective.

The sprint retrospective is usually the last thing done in a sprint. Generally it is done immediately after the sprint review. The entire team, including both the scrum master and the product owner should participate. I like to schedule retrospectives for up to an hour, which is usually quite sufficient. However, occasionally a hot topic will arise or a team conflict will escalate and the retrospective could take significantly longer.

Although there are many ways to conduct a sprint retrospective, my recommendation is to conduct it as a start-stop-continue meeting. This is perhaps the simplest, but often the most effective way to conduct a retrospective. Using this approach each team member is asked to identify specific things that the team should:

Start doing
Stop doing
Continue doing

The scrum master can facilitate this meeting by asking everyone to just shout out ideas. The scrum master can go around the room asking each person to identify any one thing to start, stop or continue.

After an initial list of ideas has been brainstormed, teams will commonly vote on specific items to focus on during the coming sprint. At the end of the sprint the team should review the list of things selected for attention in the prior retrospective.

Finally document all the lessons learned and share with relevant stakeholders.

This meeting should be done on Monday morning before the start of any sprint or a release. This meeting should be followed by sprint planning or release planning meeting.

7.14. Peer review

- Develop standard code review and test case checklist. So that reviewer keep the items in mind when he is reviewing
- Review should keep the checklist before him while conducting review
- Every code or test case should be reviewed by peer.
- Three purpose of review: a- Refining work, b- other person getting insight what is happening in team c- new people learning

- Code review should be peer developer
- Test cases should be reviewed by ticket originator
- User stories should be reviewed by ticket originator

7.15. Change Request Process

- Add the request in product backlog. (Originator)
- Assign the request (ticket) to current release (PO/PPO)
- Review the work complexity, feasibility, effort waste, impact on other aspects of functionality (Scrum Team)
- Access the impact on schedule, quality and functionality (Scrum Master)
- Tradeoffs between existing work items of sprint and new work items (PO)
- Commit only which is possible with little stretch. Do not commit impossible or where team need to work more than 20% of their capacity. (Scrum Team)
- If change is huge then complete sprint can be cancelled and sprint re-planning need to be done.
- Update the sprint backlog (SM)
- Followed by development process

8. Velocity

1st Sprint: 75 SP in 97 Hours (.77 SP/Hours). Out of 218 hours team worked only 97 hours on development work.

9. Reports

- Effort burn down chart (effort tracking within sprint)
- Product burn down chart (efforts and story points)
- Release burn down chart (efforts and story points)
- Separately captures efforts for development, maintenance and technical spikes.
- Report the velocity in terms of SP/Hours and SP/Sprint

10. Process of Doneness

This process will help stakeholders knowing % completion (earned value) rather than subjective terms like WIP, Cancelled, Close, etc.

- I suggest following ticket life-cycle and weightage for measuring the doneness of any work items in squish.

Step No.	Step Name	%	Cumulative %
•	Impact Analysis	5%	5%
•	User Story	5%	10%
•	Review of user story	2%	12%
•	Acceptance Test Case	5%	17%
•	Review of acceptance test case	2%	19%
•	Coding	45%	64%
•	Code Review	5%	69%
•	Unit Testing	10%	79%
•	QA Build & Deployment	4%	83%

•	QA Testing	10%	93%
•	UAT Build & Deployment	4%	97%
•	UAT	3%	100%
•	Product Build & Deployment		

- It can be mapped to customer templated provided supplied life-cycle of each work item.

Step No.	Step Name	%	Cumulative %
•	Requirement understanding/ Impact Analysis	10%	10%
•	IAD/ Req Review	5%	15%
•	Unit Test Case (UTC) Preparation	10%	25%
•	UTC Review	5%	30%
•	Coding	40%	70%
•	Unit Testing	10%	80%
•	Code Review	5%	85%
•	Code Rework	15%	100%

11. Trainings

- Time to time before including any new dev or test team member in scrum conduct following basic trainings
    - CBMS
    - Scrum processes
    - Ground rules of team work

12. Guidelines

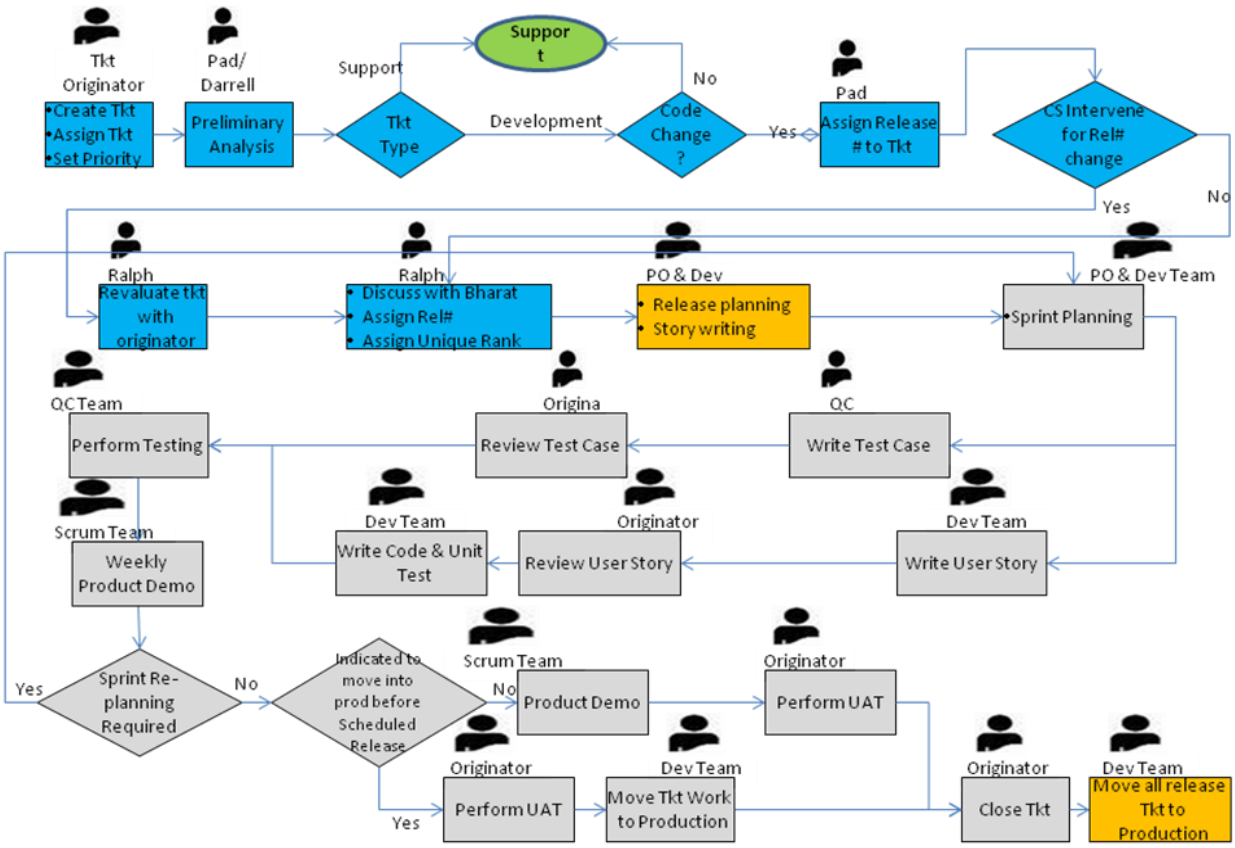
- Till velocity is not benchmarked, work items commitment for any sprint should be based on complexity and team's gut feel.
- After velocity is benchmarked. Number of user stories and velocity should be used to commit the work items.
- Developers must write user story and keep refining it based on the need. Take the approval of user stories from proxy product owner
- Proxy product owner must be available at the time of release and sprint planning. Identify some overlapping hours for release and sprint planning.
- Retrospective meeting is must after each sprint and before next sprint planning. Avoid doing this in same day.
- Tester must write test cases and take approval from proxy product owner.
- Do not assign any work items to developers. Scrum master should create an environment and self disciplined team so that team picks up the work items willingly.
- Do not put items in sprint backlog without tradeoff of existing items.
- Keep support tickets on separate track
- There should be only one proxy product owner for all requirement clarification purposes so that the team does not waste time in running after many people.
- Proxy product owner should assign unique priority (squish rank) to the work items in product backlog (squish release backlog)

- Do not take regression testing away from development work. It should be part of sprint planning and scrum team should own the delivery of quality product
- Determine team velocity in next 3-4 sprints, so that team know how much work (story points) can be committed for any particular release
- Try to keep number of development hours in any sprint constant.
- Do not change team members or introduce team members frequently it affects velocity by some unknown coefficient.
- Team members should be aware about who is doing what work. Team should take responsibility not the individuals. Team need to understand that we are responsible for workable product not for lines of code. They should help each other, even if they think that their work item is done.
- User story should be calculated for project management activities like reporting etc.
- When counting user stories for any ticket take care of integration user stories also.

13. Notes

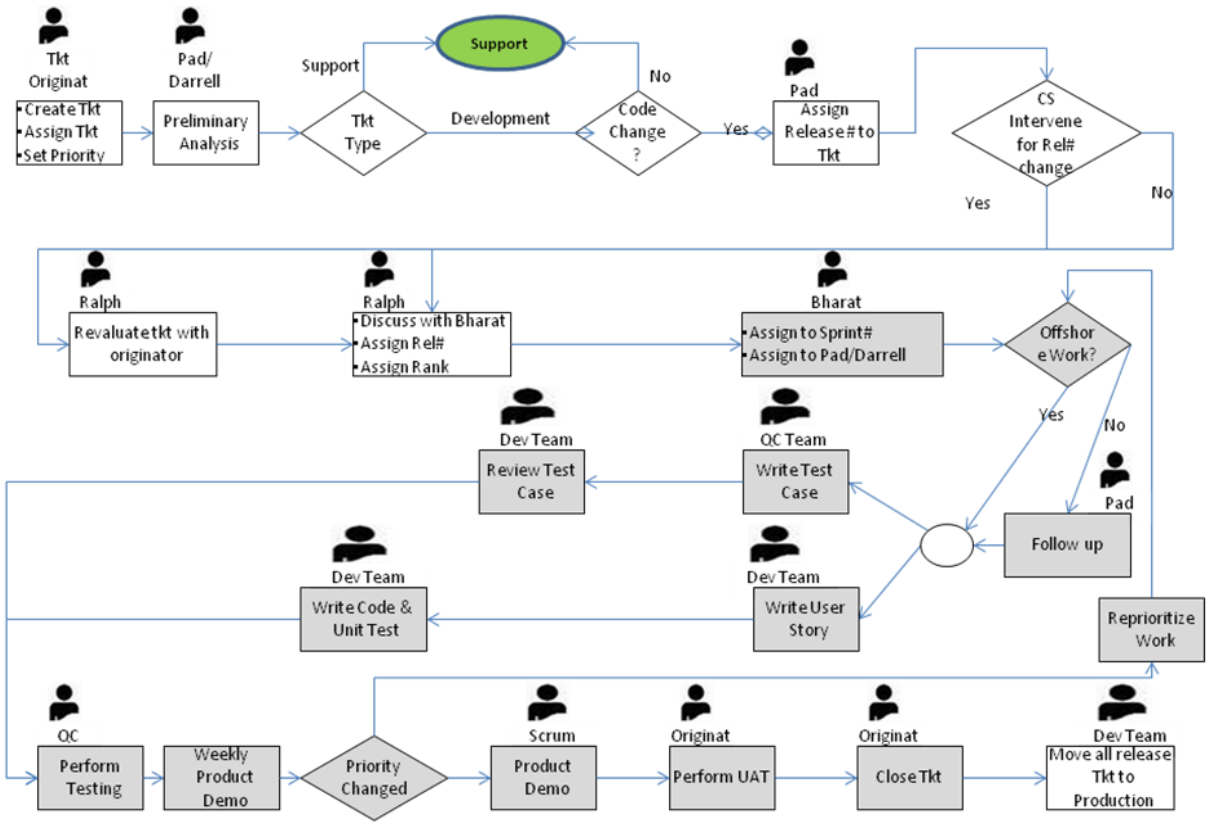
- At present in the absence of velocity we will commit based on team's gut feel. Estimate may not be true and it is possible that 3-4 sprint spill over will happen. After 3-4 sprint it can minimized to a great extent.
- Today big reason for spill over is unknown-velocity, requirement change, support tickets and not capturing efforts of support tickets.

14. Proposed Development Workflow



Development: Proposed Workflow

15. Existing Development Workflow



Development: Current Workflow

End of report