Description: 젰࢜져࢜Ø

TFS Guidelines for BAs

Version 1.0

|  |  |
| --- | --- |
| **Work Request (PPM) Number:** |  |
| **Authored By:** | **Ahmed Binafif** |
| **Author’s Role:** | **Business Unit Systems Analyst** |
| **Author’s Department:** | **DTE Business Analyst Team** |
| **Last Updated:** | **10/06/2014** |

**Release Team Cycle**

**DTE Release Cycle is made up of the following activities:**

* Planning and Prioritization – 2 week duration
* Release Backlog Creation – 2 week duration
* Release Backlog Refinement – 3 week duration
* Scrum Sprints – 12 week duration (Four 3 week sprints)
  + Sprint 0 – initial sprint of Release Cycle that has 50% allocation toward User Story refinement, and 50% allocation toward User Story execution of carry-over Epics from previous quarter and Unplanned Urgent Epics
  + Sprint 1, 2, 3 – allocated timeframe to execute User Stories in Release Backlog
* Go-Live Regression / Business Acceptance Testing – 12 days
* Go-Live Prep – 3 days
* Release Calendar will have two planned Production Go-Live weekends during a quarterly cycle, targeted at the end of sprint 1 and sprint 3.
* Go-Live weekend at the end of sprint 1 will contain:
  + Requests prioritized due to business needs (i.e. client commitment or compliance issue) by Executive Sponsors to deploy to production at the sprint 1 Go-Live
  + Requests prioritized into the Release Cycle that the Product Owner and scrum team feel can be completed during sprint 1 and can be targeted to deploy to production at the sprint 1 Go-Live
* Go-Live weekend at the end of sprint 3 will contain:
  + Requests that were completed during sprint 2 and 3 of the Quarterly Cycle.

**TFS Work Item Type Hierarchy**

Machine generated alternative text:
EPiC 
(when applicable) 
Feature 
(When needed) 
Backlog I 
Task 
Initiatives that are sufficiently substantial in scope and cost to warrant analysis and understanding 
of potential ROI. These are typically projects or portions of a large program. 
Intermediary work item type for the breakdown of Epics into smaller components when it would add 
value to do so. On an interim basis only it can be used as a roll up Of backlog items. 
Describes something that can delivered and evaluated. Includes user stories, non-functional 
requirements, spikes, and other types. May or may not be a breakdown of Epics and/or Features. 
A breakdown of a Backlog Item into activities required to complete it. A task should be assigned to 
a single person. 

**Epics**

* Each Release Item (PPM Request) will be represented in the DTE Product Backlog as an Epic. The norm is for there to be a one-to-one relationship between PPM request and Epic.

* During Intake, an epic will be created to host the scope analysis (Please refer to the example template in Epic # 42933).

* If the Intake and scope analysis was before or during the transition process, an epic will be created for the PPM work request with a short description, and the scope document shall be attached to the epic.

* When the execution of a Release Item results in dividing the Go-Live deployment of a PPM request into multiple Go-Live deployments, an Epic will be created to represent each Go-Live. User Stories associated with the PPM request will be children of the Epic representing when the User Story is Go-Live deployed.

* When multiple Epics are created for a Release Item, the Epic name convention should add an indication of the Go-Live deployment: “<PPM #> - <PPM Short Description> – Go-Live <mon/day/yr>”

* *Future consideration – use the* ***Feature Work Item*** *to represent different phase Go-Lives, therefore maintaining the one-to-one relationship between PPM request and Epic. Please refer to TFS Work Item Hierarchy above.*

* The Title of the Epic should be the following format: “<PPM #> - <PPM Short Description>”
  + Example: PPM 292122- DTE AP3 Large File Sizes Images

* Production Support Requests create an epic using the following format:
  + “Prod Support <PPM #> - <PPM Short Description>”
  + Example: Prod Support PPM 332229 - Claim transfer tool - Update fields copied

* Epic States in **Product Backlog**: (Creating a Scope Epic)

|  |  |
| --- | --- |
| **New** | Epic Created in TFS |
| **Active** | Business Need Information authored by Intake BAs (Scope information). Preliminary estimates being gathered |
| **Defined** | Estimates available. Intake Complete. Epic ready for prioritization |
| **Removed** | Epic has been cancelled by Product Owner |

* Epic States in **Release Backlog**: (picked up for user stories creation and refinement)

|  |  |
| --- | --- |
| **Active** | Scrum BAs adding detailed business need to Epic, one or more User Stories still being refined |
| **Defined** | Epic approved and all User Story refinement complete for sprint planning. This includes all tasks (BA / Dev / QA) |
| **Delivered** | All User Stories ACCEPTED at Sprint Reviews |
| **Removed** | User Story has been cancelled by Product Owner |

* In TFS, the **Priority Field** of the Epic will be used to specify these for Go-Live Statuses:
  + Priority “1” - Executive Sponsor Commitment to Deploy Sprint 1 Go-Live or an Off-Cycle Go-Live
  + Priority “2” - Product Owner and Scrum Team Target to Deploy Sprint 1 Go-Live
  + Priority “3” - Executive Sponsor Commitment to Deploy Sprint 3 Go-Live
  + Priority “4” - Epics that contain technical User stories in a Release Backlog – User Stories that need to complete during the Release Cycle, but do not represent business requests (i.e. Code Build or Code Stream Merge User Stories)

**User Story**:

* The Title of User Stories associated with an Epic should be the following format:

“<PPM #> - <User Story Number> <Short Description of User Story>"

* Example: PPM 292122- 1.0 - Download Large File Sizes Images From AP3 DTE Website

* User Story States in **Product Backlog** ( if a user story is created during Scope intake)

|  |  |
| --- | --- |
| **New** | Initial User Story being authored by Intake BAs |

* User Story States in **Release Backlog** (when an epic is prioritized for a release - Scrum Supply Work)

|  |  |
| --- | --- |
| **New** | User Story / Acceptance Criteria being authored by Scrum BAs |
| **Approved** | User Story / Acceptance Criteria ready for Task Refinement, it has been reviewed with business stakeholders and approved by them. |
| **In Analysis** | User Story Task Refinement in progress |
| **Ready** | User Story refined to point that it is ready for sprint planning |
| **Removed** | User Story has been cancelled by Product Owner |

* User Story in **Release Team Sprint Backlog** (Being worked during the sprint after being committed)

|  |  |
| --- | --- |
| **Committed** | User Story not being worked but scrum team committed to take it during Sprint Planning |
| **In Progress** | User Story actively being worked by IT-DEV or RULES |
| **In Progress** | All  IT-DEV and RULES tasks ready for test, but User Story not being tested |
| **In Progress** | QA test in progress |
| **In Progress** | All QA tests PASSED, but BA review not complete |
| **DONE** | BA reviewed test results and claimed DONE, User Story targeted for sprint review  (In Template – status indicates PO accepted) |
| **DONE** | PO and stakeholders accepted DONE status at sprint review  (Not in Template) |
| **Removed** | User Story has been cancelled by Product Owner |

* In TFS, the “**Stack Rank**” field will be used to represent User Story Priority:
  + Stack Rank of “1” - HIGH – must be added to the next sprint backlog
  + Stack Rank of “2” - MEDIUM – Tier 2 priority to consider to be added to the next sprint backlog
  + Stack Rank of “3” - LOW – Tier 3 priority to consider to be added to the next sprint backlog
  + Stack Rank of “4” - **BLOCKED** – not a candidate to be added to the next sprint backlog (BLOCKING issue should be documented in BLOCKED field of User Story)

**Tasks**:

* Task States in **Release Team Release Backlog**

|  |  |
| --- | --- |
| **New** | Task being authored or has been authored by Scrum Team |
| **Removed** | Task has been cancelled by Scrum Team |

* Task States in **Release Team Sprint Backlog**

|  |  |
| --- | --- |
| **New** | Execution of task has not started |
| **Active** | Task being executed |
| **Closed** | Task execution complete |
| **Removed** | Task has been cancelled by Scrum Team |

* When working tasks from a sprint backlog, the task priorities should be consistent with the priorities of the User Story that the task is associated with (work tasks associated with User Stories of PRIORITY HIGH first,…..)**.**

* During a Sprint, prior to each daily stand-up, a task owner needs to update the following fields for User Story tasks that are actively being executed:
  + Remaining Work – update with the estimated hours remain to complete execution of the task
  + Completed Work - Use this field for the hours that you have completed on a task.

* **BA Standard Task Template**
  + BA Support /Review
    - **Estimate**: Depends on size of User Story
    - **Purpose**: All support tasks the BA will encounter throughout the Sprint. ITQA/Dev Support, answering requirements questions, reviewing test results, etc.
    - **Owner**: BA only

**TFS Template for Bug Work Items**

* When entering a defect as a Bug Work Item, the following fields of the Work Item should be completed:
  + **Repro Steps** – Steps to reproduce the defect, with the unexpected test results included
  + **System Info** – any information pertaining to operational environment (i.e. software version, workstation environment)
  + **Test Cases** – link to test case used to expose the defect
  + **Attachments** – screen shots of unexpected test results if helpful

* Defect States:

|  |  |
| --- | --- |
| **New** | * + New Defect being triaged for necessary corrective action.   + “Assigned To” field should be a BSA |
| **Active** | * + A corrective action has been documented.   + “Assigned To” field should the IT-DEV resource.   + The defect stays in this state as it is being worked by IT-DEV, IT-QA, and other testing teams   + Once the defect is fixed and the code is promoted to Model, the developer shall change the “Assigned To” field to the BSA who created the defect for validation. |
| **Closed** | Defect has been fixed and tested |
| **Removed** | Cancelled. Not a defect |

* The Reason field should be used in the Closed and Removed states:
  + If the State is **Closed**, the **Reason** shall be Verified
  + If the State is **Removed**, the **Reason** shall be one of the following:
    - As Designed
    - Cannot Reproduce
    - Duplicate
    - Obsolete

* Defect Severity

|  |  |
| --- | --- |
| **1 – Critical** | show-stopper/blocking issue |
| **2 – High** | impacts operational functionality. |
| **3 – Medium** | has an available work around |
| **4 – Low** | nice to have |

* Bug Annotation
  + During the execution to resolution of a defect, information such as clarification of root cause and status of correct action should be captured as Comments in the TFS Bug Work Item.
  + When a defect corrective action is promoted to Test, an entry should be added at the top of the “Repro Step” Field that explains the Resolution of the corrective action.

**Defect Tracking During Sprint**

* During sprint execution, the scrum team will be resolving issues associated with Test Case execution failures as they progress a User Story to DONE.

* Best practice is to capture the results of each test case execution as an attachment to the Test Case.

* **Deferring defects from a sprint** :
  + The scrum team will collaborate with the Product Owner to defer investigating or resolving an issue raised during User Story testing until after the active sprint, and to claim User Story DONE during that sprint, even with the open issue.
  + When the scrum team and Product Owner claims a User Story is DONE with unresolved issues existing from test, those issues must be documented in the TFS Project as Bug Work Items in the Release Backlog.
  + The Bug Work Item should be linked as a Child to the Epic parent of the User Story the issue is associated with.
  + When creating a Bug Work Item for an issue to be deferred from a sprint, the Bug Work Item Title should follow the following formats:
    - <PPM #> <Description>
    - The Test Phase field should be set to “QA”

* **Creating Bug Work Items for defects resolved in a sprint**.
  + For internal QA metrics, the QA team may opt to create bug work items for test case execution failures even when the issue is investigated and resolved during the sprint.
  + These QA internal Bug Work Item artifacts should be isolated from Backlog and sprint management by setting the Test Phase field to “Intra-sprint”.

**Defect Tracking During UAT, Regression & Warranty**

* All defects identified should be entered to the TFS Project as Bug Work Items in the Release Backlog.

* If the defect can be associated with a specific Epic developed for the Go-Live being tested, the Bug Work Item should be linked as a Child to the Epic.

* When creating a Bug Work Item, the Title should follow the one of these formats:
  + <PPM #> <Description>
  + or
  + Unattached – <Description>

* If the Bug was detected during UAT, the Test Phase field should be set to “UAT”
* If the Bug was detected during Regression, the Test Phase field should be set to “Regression”.
* If the Bug was detected during Warranty, the Test Phase field should be set to “Warranty”

* Defects will be reviewed by the Product Owner and those that are identified as “Must-Fix” will be moved to the active Sprint Backlog and become part of that sprint.

* Bug Work Items that are identified as a must fix for a specific Go-Live date, the Title should be edited to the following format:
  + <PPM #> <Description> - (mm/dd Go-Live)

* These “Must-Fix” defects will be implemented in the Go-Live Release code stream and tested directly in MODEL. A User Story will be added to the Release Backlog to merge these “Must-Fix” changes into the future Release code stream.

**Processing Defects when closing the Release Backlog**

* The Release Backlog will be Closed at the end of the warranty period, and all Epics deployed to Production will also be Closed.

* All open Bug Work Items in the Release Backlog will need to be processed in one of the following ways:
  + If the Product Owner feels the Open Bug Work Item should be considered for resolution in a future Release Cycle, a Production Support Intake Request should be generated
  + If the Product Owner feels no further action should be taken on the Open Bug Work Item, the Open Bug Work Items should be Closed

**Annotation Guidelines:**

* BA shall annotate all updates regarding the work item in the History Tab.
* BA shall annotate all peer reviews of the epic/user story that they are working on in the History tab.
* As user stories transition to DONE state, the BA shall arrange an advance demo with operations and document whether a decision was made to have a demo or not.
  + If the decision was to have a demo, then the BA shall note the date and time of the demo, as well as a post demo status and feedback