Bug Severity Classification

Documentation ID	Ver.	Date	Description	Changed by
DID007	1.0	11/4/16	Bug Severity Classification	Rami

Stage	Role	Name
Author/Lead	QA Lead	Rami
Reviewer	Release Manager/ Product Manager/ Engineering Lead	
Approver	Director QA	

Severity and Priority

? Severity describes the impact on the customer

- ? Best judgment should be used to determine the Severity according to the definitions herein
- ? It is allowed to change the severity of a defect if new information becomes available
- ? With detailed analysis, the Severity can “go” up or down

? The QA/ENG/Project manager (basically anyone) can alter the Severity. Upon conflict PM has the last call

? Priority manages the urgency & order of handing defects
?
Priority with severity can be used to determine the immediacy of repair. Usually priority serves to prioritize among defects with different or same Severity based on the likelihood that a customer will encounter the problem. During NPI the below and similar can dictate:

-Test plan execution – i.e. a Test Stopper (TS) -Release Criteria – i.e. Show Stopper or Release Stopper (RS) -Beta customer testing – i.e. Beta Stopper (BS) -Demo or Marketing Requests – Demo/Mkt request (Demo)

Priority is not part of the release criteria
?

Defect Severity background

- ? Line up with ANSI/IEEE Std 729-1983 Glossary of Software Engineering Terminology
  - ? First, it must be determined if the defect resulted in a system failure. ANSI/IEEE Std 729-1983 defines a failure as, "The termination of the ability of a functional unit to perform its required function."
  - ? Second, the probability of failure recovery must be determined. ANSI/IEEE 729-1983 defines failure recovery as, "The return of a system to a reliable operating state after failure."
- ? There are 5 levels industry standard for defects. It is common practice to include at the lower level enhancements request.
- ? 5 level Severity (ANSI terms for comparison)
  - ? S1 Catastrophic (Critical)
  - ? S2 Severe (Major)
  - ? S3 Moderate (Average)
  - ? S4 Minor (Minor)
  - ? S5 Cosmetic (Exception) + Enhancement

Defect Severity Definitions

?S1 Catastrophic :? Reasonably common circumstances causes the entire system to fail, or a major subsystem to stop working, or disrupting oth er devices in the topology, and there is no workaround. The problem is persistent ?

S2 Severe: ? Important functions are unusable, and there is no workaround. The issue is disruptive but does not inhibit the usage of a main flow, in a production environment

?S3 Moderate : A failure under unusual circumstances, or minor feature doesn't work at all, or the failure has a low–impact with workaround. ?

S4 Minor : A failure under very rare circumstances, with possible self recovery. Workaround exist, and the performance impact is tolerable

?S5 Cosmetic :? There is no real impact/effect on the system functionality

? And or Enhancement
? A request for new functionality or for an improvement to an existing feature that is not part of any PRD.

Defect Priority Definitions

? P1 Blocker :Resolve Immediately – Very likely that a customer will encounter the problem. Further development and/or testing cannot occur until the defect has been repaired. The system cannot be used until the repair has been made. ?

P2 Critical : Give High Attention - Somewhat likely that a customer will encounter the problem. The defect must be resolved as soon as possible because it is impairing development/and or testing activities. System use will be severely affected until the defect is fixed.

?P3 Major: Normal Queue - Risk is low that a customer would encounter this problem. The defect should be resolved in the normal course of development activities. It can wait until a new build or version is created. ?

P4 Minor : Low Priority - Risk is very low that a customer would encounter this problem The defect is an irritant which should be repaired when feasible.

Defer: The repair can be put of indefinitely. It can be resolved in a future major revision or not resolved at all. This is ANSI additional level – similar for the “known for version” state. WILL NOT BE USED

Defect-Examples by Severity

- ? S1 Catastrophic
  - ? Example 1: No video / audio / data in a commonly used scenario
  - ? Example 2: Unable to make calls in a commonly used scenario
  - ? Example 3: Uncontrolled system restart at least ones a day
  - ? Example 4: production outage - complete outage for one service or one reproducible scenario
- ? S2 Severe
  - ? Example 1: Audio problems (metal voice, high pitch tone)
  - ? Example 2: Video problems (low frame rate, bad video)
  - ? Example 3: Uncontrolled system restart of more than ones every week
  - ? Example 4: partial production outage - partial outage for one service or single scenario
  - ? Example 5: Recoverable data loss or corruption
  - ? Example 6: Performance degradation that impedes usage

Defect Examples cont.

- ? S3 Moderate
  - ? Example 1: IOT problems with specific endpoint
  - ? Example 2: An isolated issue, that causes controlled/ manageable impact on small and limited number of end users.
  - ? Example 3: Major visible defect like incorrect error message, or errors in UI.
- ? S4 Minor
  - ? Example 1: A typo in a user menu that is visible in a normal usage
  - ? Example 2: A misleading Translation to another language
- ? S5 Cosmetics
  - ? Example 1: Typo in a debug log message
  - ? Example 2: Minor typo in a UI error message that is not commonly shown to a regular user ? Enhancement

? Example 1: A valuable request from customer to improve/change something in the product, but the request did not make it to the product requirements at this stage

The Gain in Early Finding Faults

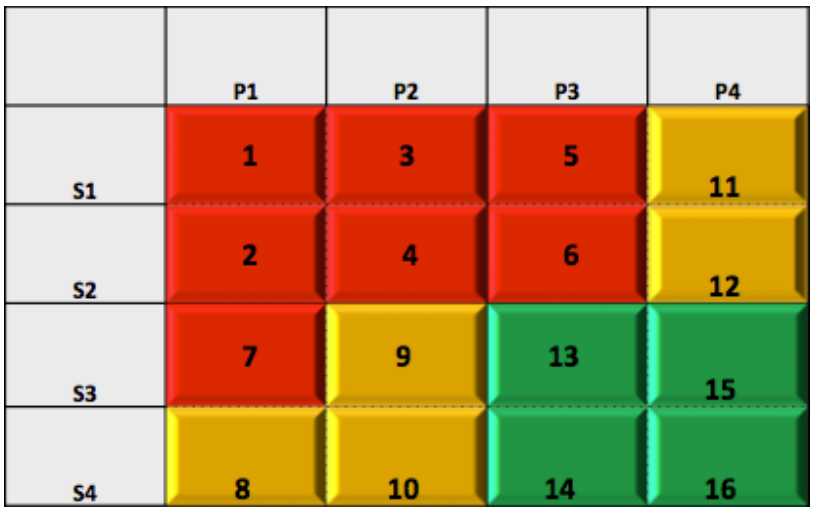
It is commonly believed that the earlier a defect is found the lower will be the cost to fix it.

The table shows the cost of fixing a defect depending on the stage it was found. For example, if a problem in the requirements is found only

during post- release, then it would cost 10–100 times more to fix than if it had already been found during the requirements review.

		Time detected				
		Requirements	Architecture	Construction	System test	Post-release
Time introduced	Requirements	1×	3×	5–10×	10×	10–100×
	Architecture	-	1×	10×	15×	25–100×
	Construction	-	-	1×	10×	10–25×

Severity/Priority Handling Order View



Order of handling defects to ensure less customer impact down to less likely to be seen at customer sites.

S1-S2 / P1 – P3: Critical or high customer impact and likely or somewhat likely that a customer will encounter the problem.

S1-S2 / P4: While impact is high, risk is very low that a customer would encounter this problem
S3 / P1: Medium customer impact and likely that it will be seen by customers (e.g., would result in multiple escalations)
S3 / P2: Medium customer impact and medium risk that a customer will encounter the problem

S3-S4 / P3-P4: Isolated customer impact, Business can tolerate deferral

S4 / P1-P2: Isolated user impact but high business impact if not fixed. This defect categorisation should be rare. Most S4 defects will be P3 or P4.

Notes from this meeting:

Jira Priority

- Stories
  - Ownership: Product Owner
- Epics
  - Ownership: Product Owner
- Bugs
  - Ownership: CS & CSM for field cases
  - Ownership: Product Owner for Internal QA

CRM Priority (Zen Desk)

- This field will be removed and no longer used.

Defect Severity (only applicable to Bug)

- Ownership of field (internal testing): QA
- Ownership of field (field case): CS
- Can be changed based on more information based on analysis.
- S1-S5
- Impacts quality grade for released product

Blocks (Gate Blocker)

- Beta/GA/Test, etc.

Code Red

- Creation of dedicated Endpoint & Infrastructure teams
- SLA process will be handled by Scrum masters of both teams
- Risk to SLA's will be communicated by the respective Scrum masters
- "Code Red" as a label will no longer exists

Severity/Priority Handling Order

- This will be formally adopted and used to clean up the current items in the system (all 16 levels)