|  |
| --- |
| **TEST PLAN** |
| **The Elder Scrolls V : Skyrim**  **Version v1** |

|  |
| --- |
| Xynan Lee  8-3-2017 |

**Document Control**

**Document Detail**

|  |  |
| --- | --- |
| Title | Test Plan for The Elders Scrolls V : Skyrim |
| Version | v0.1a |
| Date | 16 July 2017 |
| Electronic File Name | SkyrimTestPlan\_V1.docx |
| Electronic File Location | https://drive.google.com/drive/folders/0B1CRg3BMYQ97MU0zNFEwUHlENmM |
| Author | Lee Choon Meng |
| Contributors | Michael Ooi |

**Document Change Control**

|  |  |  |  |
| --- | --- | --- | --- |
| Issue Date | Version | Details | Author |
| 3 August 2017 | v1 | Full document completed | Lee Choon Meng |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Referenced Documentation**

|  |  |  |
| --- | --- | --- |
| No. | Document Name | Document File Location |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

|  |  |
| --- | --- |
| Title | Page |
| 1. **Executive Summary** | **4** |
| 1. **Introduction**    1. Project Objectives    2. Document Objectives    3. Related Documents | **5**  5  5  5 |
| 1. **Test Scopes**    1. Test Objectives    2. Test Lifecycle    3. Test Levels    4. Test Types    5. Features to be Tested    6. Features Not to be Tested | **6**  6  6  7  8  9  10 |
| 1. **Resources and Staffing**    1. Test Environments    2. Testing Tools    3. Roles and Responsibility       1. Manpower Staffing       2. Tester Grouping    4. Training       1. Training Objectives       2. Training Plan | **10**  10  14  14  15  15  18  18  18 |
| 1. **Test Approaches**    1. Test Protocols and Standards    2. Test Criteria       1. Pass and Fail Criteria       2. Entry and Exit Criteria       3. Suspension and Resumption Criteria    3. Test Measurements and Metrics       1. Testing Measurements       2. Testing Metrics    4. Documentation       1. Test Plan Identifiers (ID)       2. Test Case    5. Communication and Status Report       1. Test Deliverables          1. Test Report          2. Test Summary          3. Defects Report and Tracking       2. Test Progress Report    6. Risk and Mitigation    7. Change Control Management       1. Document Control Form       2. Change Request Form | **20**  20  20  20  21  22  22  22  23  24  24  27  27  29  29  29  30  32  34  35  35  37 |
| 1. **Requirement Traceability Matrix** | **39** |
| 1. **Approval** | **40** |
|  |  |

**1. Executive Summary**

This document is created as a guideline for the development team and the testing department to assess, test, iterate and improve the development processes, testing processes. The primary purpose of the document is to make sure the Quality Assurance processes and Quality Control test are delivered efficiently and accurately. The scope of testing covers all possible testable units and areas such as functional, performance, compatibility test etc. The software development life cycle used is **incremental** model. Test plans and test cases will be derived from this test strategy. As a conclusion, the Quality Assurance, Quality Control and the development team must comply with all of the procedures defined in this document in order to deliver the product as expected.

**2. Introduction**

**2.1 Project Objectives**

The objectives of the project create a unique and more diverse game world than Oblivion. The team also sought to create a more diverse world that feel topographically unique from each of the holds, which are small teams that are responsible for creating a more unique world. The development team also wanted to reflect the socioeconomic background of NPCs by making some places more developed and wealthy while others are poorer and rural. Overall, the development team was trying to achieve a better more immersive world with more realistic elements and emphasized more on players’ discovery and exploration.

**2.2 Document Objectives**

This particular document is written to ensure the Quality Assurance and Quality Control testing that would be undertaken throughout the project performed correctly according to the defined conditions in the document. This document serves to communicate the testing approach to the Leads and testers. This document will cover the testing techniques, testing schedule, scope and tests that need to be carried out alongside will their details. This document will influence other documentations and task such as test cases, and evaluations and so on.

**2.3 Related Documents**

|  |  |
| --- | --- |
| **Document Name** | **Link** |
|  |  |
|  |  |

**3. Test Scopes**

**3.1 Test Objectives**

In reference with the project objectives, the primary objectives of all test activities are to:

* Ensure the development team delivers its efforts without compromising the production timeline and efficiency.
* Ensure the core mechanics of the game is delivered without any issue.
* Ensure all planned mechanics and features are implemented and functional.
* Ensure all planned mechanics and features deliver expected behaviours and results.
* Ensure all planned mechanics and features fulfil the business requirements of the project.
* Eliminate bugs and minimize bug reproducibility as much as the team can without compromising the production timeline and efficiency.
* Ensure the development builds go through each development phases are functional.
* Identify and fix any bug and error before the build is carried to the next phase.

**3.2 Test Lifecycle**

The test lifecycle to be used in this project is based on **incremental** development model of testing. Due to the immense scale of the game, the minimum viable product must be developed first and foremost to get the basic concept and the gameplay out first. Additional features and downloadable content patches will be added upon the existing builds if the existing builds passes all of the tests drafted for it. A regression test will be conducted if major changes made in the game or play tests are around the corner.

The test techniques that will be mainly used in Black Box testing are:

* **All-pairs testing** because of the possible pairings of the situation that can happen such as player equip sword, player equip staff, player equip magic etc.
* **Equivalence partitioning** because of the sheer scale of the game with so many props and objects. It is easier to group them together according to their common traits to conduct tests.
* **Classification tree method** can also be used to test the different races, different category of skills, perks to make testing easier.
* **State transition testing** is used to test the decisions made by the player and the actions made by the player altering the world condition.

The test techniques that will be mainly used in White Box testing are:

* **Control flow testing** to test the flow of execution of the test item to ensure the test item operates according the desired flow.
* **Data flow testing** to test the correct data is passed to the correct destination and to ensure there are no false passing and unexpected data passes.
* **Branch testing** to ensure all possible branches are executed at least once to ensure there are no unreachable codes.
* **Decision coverage** to test the decisions and actions made by the player impacts the world condition correctly as expected.
* **Path testing** to test the different questlines and their possible endings.

**3.3 Test Levels**

The following are the test levels which will be conducted throughout the project.

* **Unit Tests**

Unit tests are tests drafted to test smallest, independent and testable part of the source code. They are conducted by the developer or developers of that unit. Unit tests must be performed in all units before the units are integrated. Developers have to draft well-defined unit test plans and unit test cases to execute unit testing perfectly. Unit tests are carried out to ensure the tested units are functional and display expected results and behaviours.

* **Integration Tests**

Units passed the unit tests will be integrated to create more complex results and behaviours. The integrated units now must undergo integration tests. Integration tests are drafted to test the interactions between one or more units and their outputs in various scenarios whether these units are able to perform as expected when integrated together. Integration tests can reveal defects and errors related to functional, requirement and performance levels. Integration tests must be carried out before the integrated units and components are linked together to form a system.

* **System Tests**

After the integrated units and components passed the integrated tests, they are all linked together to form a system. System tests are carried out to validate the whether the system meets its requirements, objectives of the project and whether the whole system perform as expected.

**3.4 Test Types**

* **Functional Tests**

Functional tests are tests carried out to ensure that each function or functional unit executes in accordance with the requirement specifications. Functional tests are a form of black box testing is not conducted in reference with the source code of the system. These tests are drafted to verify the user interface, APIs, databases, security, client/ server applications and their functionalities.

* **Performance Tests**

Performance tests are created to identify the efficiency and response time of the application. Efficiency of the application is measured in terms of CPU usage, RAM usage and storage usage, the lower the usages, the faster the response time, the more efficient the application is.

* **Compatibility Tests**

Compatibility Tests are carried out to validate the compatibility of the application with different hardware, software and environment configurations to let the end users know what systems the application can run on without actually purchasing or installing the application in their personal computers.

* **Component Interface Tests**

Component interface tests are formulated to check the handling of data passed between various units or components which are the background processing that are not directly visible to users. These tests are conducted to ensure the correct data are passed to the correct units or components and ensure there are no unusual data exists.

* **Acceptance Tests**

After the system is thoroughly tested, the quality assurance team must test the system for its quality in reference with existing test scenarios and test cases. Acceptance tests examine the system is different angles, from cosmetics looks to internal functioning to ensure the system comply with the legal requirements and client’s needs. Acceptance tests can come in following forms:

* + User Acceptance Tests

User acceptance tests are tests that carried out by the end user or the client by the software is accepted.

* + Operational Acceptance Tests

Operational acceptance tests are formulated to ensure that all the processes and steps are all in place so that the software can be used and maintained easily.

* + Contract and Regulation Acceptance Tests

Contract and regulation acceptance tests are carried out to ensure the application abides all the government, legal and safety standards and regulations.

* + Alpha Tests

Alpha tests are tests performed towards the end of the development to ensure the quality of the software as a whole. The tests can be either simulated or tested by clients or test teams from the development group where is tested from the client’s point of view.

* + Beta Tests

Following Alpha tests are Beta tests are tests conducted outside of the development test and are released to limited public audience. Beta tests are carried out to see if the software appeal to the public world.

* **Regression Tests**

Regression Tests are big tests that involves retesting all of the features, units and functions to ensure that recent changes in the system does not affect the existing features and their functionalities and does not create abnormalities in the system.

**3.5 Features to be Tested**

The following are the feature categories to be tested are ranked from top most priority to be tested which they can severely impact the gameplay to the lowest most where they do not affect much of the game itself.

|  |  |
| --- | --- |
| **Feature Categories** | **Priority** |
| Movement | **High Priority** |
| Health |
| Leveling |
| Quest (Main quests) |
| Inventory |
| Environment |
| Combat |
| Camera |
| Model | **Middle Priority** |
| Stamina |
| Mana |
| Shop Mechanic |
| Animation |
| Mount |
| UI |
| Character Creation & Customization | **Low Priority** |
| Quest (Side quests and Optional quests) |
| UX |
| Audio |
| Miscellaneous |

**3.6 Features Not to be Tested**

Due to the reason where the target application is the base game, therefore there are no features that could bypass the testing stages.

**4. Resources and Staffing**

**4.1 Test Environments**

The following are the software and hardware required for before the tests before the tests being carried out.

|  |  |  |
| --- | --- | --- |
| **Platform** | **Requirements** | |
| **Software** | **Hardware** |
| PCs | Minimum Requirements | |
| *OS* :   * Windows XP (32 bit)   *DirectX* :   * DirectX 9.0c   *Additional* :   * Internet access for Steam activation | *Processor* :   * Intel Celeron E1200 Dual-Core 1.6Ghz * AMD Athlon 64 X2 Dual Core 3600+   *RAM* :   * 2GB System RAM   *VRAM* :   * 512MB VRAM   *Graphic Card* :   * NVIDIA GeForce 7600 GT * NVIDIA Quadro FX 380 LP * ATI Radeon X1800 * ATI Radeon HD 3600 Series   *Hard Disk* :   * 6GB free HDD space   *Sound Card* :   * DirectX 9 compatible   *Network* :   * Broadband / WLAN   *Monitor:*   * 1280 \* 720 (720p) * 1920 \* 1080 (1080p)   *Keyboard and Mouse* |
| Recommended Requirements | |
| *OS* :   * Windows 7 (64 bit) * Windows 8 (64 bit) * Windows 8.1 (64 bit) * Windows 10 (64 bit)   *DirectX* :   * DirectX 9.0c   *Additional* :   * Internet access for Steam activation | *Processor* :   * Intel Core 2 Quad Q9400 2.66Ghz * AMD Phenom II X4 920   *RAM* :   * 4GB System RAM   *VRAM* :   * 1GB VRAM   *Graphic Card* :   * NVIDIA GeForce GTX 260 or higher * ATI Radeon HD 4890 or higher   *Hard Disk* :   * 6GB free HDD space   *Sound Card* :   * DirectX 9 compatible   *Network* :   * Broadband / WLAN   *Monitor:*   * 1920 \* 1080 (1080p)   *Keyboard and Mouse* |
| Xbox 360 | *OS* :   * Xbox 360 system software   *Additional* :   * Internet access * Xbox Live services | *Processor* :   * 3.2Ghz PowerPC Tri-Core Xenon   *RAM* :   * GDDR3 512MB RAM @ 700Mhz   *Display* :   * Composite video * S-Video * RGB SCART * Component (YPBPR) * D-Terminal (YPBPR) * VGA * HDMI   *Graphic Card* :   * 500Mhz ATI Xenos   *Hard Disk* :   * Detachable Hard Drives * Memory Cards * On-board storage chip * USB * Cloud Storage   *Sound Card* :   * Analog Stereo * Stereo LPCM * Dolby Digital 5.1 * Dolby Digital with WMA pro   *Network* :   * Broadband / WLAN   *Controller* :   * Xbox 360 Controller (USB wired, 2.4Ghz wireless) |
| PlayStation 3 | *OS* :   * PlayStation 3 system software   *Additional* :   * Internet access * PlayStation Network services | *Processor* :   * 3.2 Ghz Cell Broadband Engine with 1 PPE and 8 SPEs   *RAM* :   * 256MB XDR DRAM system * GDDR3 256MB video   *Display* :   * Composite video * S-Video * RGB SCART * Component (YPBPR) * D-Terminal * HDMI   *Graphic Card* :   * 550Mhz NVIDIA/SCEI RSX Reality Synthesizer   *Hard Disk* :   * Detachable Hard Drives * Memory Cards * On-board storage chip * USB * Cloud Storage   *Sound Card* :   * Analog Stereo * TOSLINK (LPCM, Dolby Digital 5.1, DTS 5.1, AAC) * HDMI (LPCM, DTS-HD Master Audio, Dolby TrueHD Bitsteam, Dolby Digital 5.1, DTS 5.1, AAC)   *Network* :   * Broadband / WLAN   *Controller* :   * DualShock 3 * DualShock 4 |

**4.2 Testing Tools**

The application is built upon Creation Engine, which is built specifically for the development of The Elder Scrolls V: Skyrim. Creation Engine is also used to conduct all white-box testing, i.e. unit and integration testing. This tool is also used for debugging the source code.

**4.3 Roles & Responsibility**

|  |  |
| --- | --- |
| **Roles** | **Responsibility** |
| Designers | Black Box testing |
| Debugging design flaws |
| Programmers | White Box testing |
| Debugging codes & application’s functionality |
| Artists | Refining models |
| Adjusting animations |
| Quality Assurance Leads | Guide the Quality Assurance Testers on how to carry out different types of testing |
| Report to Quality Assurance Manager |
| Quality Assurance Testers | Carried out the tests assigned by Quality Assurance Leads |
| Report to Quality Assurance Leads |
| Quality Control Leads | Guide the Quality Control Testers on how to carry out different types of testing |
| Report to Quality Control Manager |
| Formulate test cases |
| Quality Control Testers | Carried out the tests cases assigned by Quality Control Leads |
| Report to Quality Control Leads |
| Test Environment Team | Set up the correct environment for specific tests |
| Assist Test Environment Manager |

**4.3.1 Manpower Staffing**

For this application, 8 groups of testers will be assembled consisting of 3 groups of Quality Assurance testers and 5 groups of Quality Control testers. The table below shows the composition of the testing department.

|  |  |
| --- | --- |
| **Portfolio** | **Number of people** |
| Quality Assurance Manager | 1 |
| Quality Assurance Leads | 3 |
| Quality Assurance Testers | 15 |
| Quality Control Manager | 1 |
| Quality Control Leads | 5 |
| Quality Control Testers | 25 |
| Test Environment Manager | 1 |
| Test Environment Team Members | 5 |
| **Total** | **56** |

**4.3.1 Tester Grouping**

The table below shows the groupings done for tests assigned.

|  |  |  |
| --- | --- | --- |
| **Group Name** | **Leader** | **Testers** |
| Alpha  ID : 01 | Quality Assurance  Lead 1  (Mr. A) | Names:   1. David 2. Roy 3. Tim 4. Joe 5. Katherine |
| Bravo  ID : 02 | Quality Assurance  Lead 2  (Mr. B) | Names:   1. Adam 2. Laila 3. Jeff 4. Terry 5. Mona |
| Charlie  ID : 03 | Quality Assurance  Lead 3  (Ms. C) | Names:   1. Tom 2. Brittany 3. Max 4. Dan 5. Ken |
| Delta  ID : 04 | Quality Control Lead 1  (Mr. D) | Names:   1. Beckett 2. John 3. Kimmy 4. Demson 5. Sam |
| Echo  ID : 05 | Quality Control Lead 2  (Ms. E) | Names:   1. Aisling 2. Daniel 3. Brandon 4. Kona 5. Royce |
| Foxtrot  ID : 06 | Quality Control Lead 3  (Ms. F) | Names:   1. Ryder 2. Collin 3. Craig 4. Ramsay 5. Jim |
| Golf  ID : 07 | Quality Control Lead 4  (Ms. G) | Names:   1. Ivan 2. Chris 3. George 4. Jeff 5. Chin |
| Hotel  ID : 08 | Quality Control Lead 5  (Mr. H) | Names:   1. Jack 2. Jamal 3. Freya 4. Drake 5. Ryan |
| India  ID : 09 | Test Environment Team  (Mr. I) | Names :   1. Lee 2. Khan 3. Jane 4. Mayson 5. Quinta |

**4.4 Training**

**4.4.1 Training Objectives**

The purpose of training the testers to ensure the testers totally understand the protocols and procedures of conducting the tests that have to been assigned to them in order to avoid false tests and reports.

**4.4.2 Training Plan**

a. Training Type

The following are different types of trainings that will be carried out throughout the project.

|  |  |
| --- | --- |
| **Training Type** | **Description** |
| Project Protocols and Standards | Teaches about the necessary protocols and standards that have to abide all the time throughout the project. |
| Test Tool Usage | Teaches about the ways of using the interface and model viewers on Creation Engine and also source code editors or other applications if necessary. |
| Test Case Execution | Teaches about the necessary procedures needed when performing test cases. |
| Test Environment Settings | Teaches about the procedures and rules needed to abide when preparing test environment. |
| Test Documentation | Teaches the correct way of recording and filing documents for test plans, test cases, test summary, error reports, bug reports etc. |

b. Training Approach

The table below shows the necessary details for the trainings.

|  |  |  |  |
| --- | --- | --- | --- |
| **Training Details** | | | |
| **Training Location** | | **Testing Lab** | |
| **Training Facilitators** | **Assignments** | **Date** | **Comments** |
| Quality Control Manager | ***Attendees:***   * Designers * Artists * Programmers   ***Assigned Training:***   * Test Case Execution * Test Tool Usage * Test Documentation * Project Protocols and Standards |  |  |
| Quality Assurance Manager |
| Quality Assurance Manager | ***Attendees:***   * Team Alpha * Team Bravo * Team Charlie   ***Assigned Training:***   * Test Case Execution * Test Tool Usage * Test Documentation |  |  |
| Quality Control Manager | ***Attendees:***   * Team Delta * Team Echo * Team Foxtrot * Team Golf * Team Hotel   ***Assigned Training:***   * Test Case Execution * Test Tool Usage * Test Documentation * Project Protocols and Standards |  |  |
| Test Environment Manager | ***Attendees:***   * Team India   ***Assigned Training:***   * Test Tool Usage * Project Protocols and Standards * Test Environment Settings |  |  |

**5. Test Approaches**

**5.1 Test Protocols and Standards**

The followings are the legal standards that the product, the developers and the development procedures must abide to.

* **Entertainment Software Rating Board (ESRB)**

Identify the suitable age range for the video game for publishing the video game in Canada, Mexico and the United States of America.

* **Pan European Game information (PEGI)**

Similar to ESRB rating system, but PEGI is needed for publishing in European countries.

* **Microsoft Services Agreement**

This agreement is needed in order to publish on the Xbox360 console and Xbox Live.

* **Steam Subscriber Agreement**

The agreement to allow the developers to publish the game on Steam.

* **PlayStation Partner Agreement**

Similar to Microsoft Services Agreement, the game must comply to this Agreement in order to publish on PlayStation consoles and PlayStation Network.

* **Nintendo Development Terms of Service**

This agreement is a necessary for the game to publish on Nintendo devices.

**5.2 Test Criteria**

**5.2.1 Pass and Fail Criteria**

Each test case will be assigned a **PASS** or **FAIL** state according to two criteria:

* Total number and severity of bugs in Unresolved / Pending state in any Bug Tracker technique or software used.
* Frequency or rate of successfully executed all test requirements and obtained expected results.

Table of Bug Severity

|  |  |  |
| --- | --- | --- |
| **Severity** | **Definition** | **Allowable Threshold** |
| S1 | System crash, data loss, no workaround, legal issues | 0 |
| S2 | Operational error, incorrect or unexpected result, | 10 |
| S3 | Minor bugs | 20 |
| S4 | Incidental, ignorable bugs, cosmetic issues | 50 |

Bugs that does not occurred more that the Allowable Threshold can remain in Unresolved / Pending state for the test case and be acceptable for release.

Table of Test Scenario Priority

|  |  |  |
| --- | --- | --- |
| **Test Scenario** | **Definition** | **Minimum Pass Rate** |
| P1 – Critical | Essential to the Product | 100% |
| P2 – Important | Necessary to the Product | 70% |
| P3 – Desirable | Preferred to the Product | 30% |

A test case must pass the similar or more than their respective Minimum Pass Rate can be considered for release.

Unforeseen issues arise during testing might affect the ‘Pass / Fail’ status of the test case. The issues can be reviewed with the test teams and project executives.

**5.2.2 Entry and Exit Criteria**

**Entry Criteria**

All the following criteria must be fulfilled before the test case is accepted into the testing phase.

* All test tools and infrastructure are available for use during testing.
* The latest version of build have been deployed to the correct test environments.
* Unit tests have been passed and completed.

**Exit Criteria**

In order to complete testing phase, all the criteria below need to be achieved:

* Test Summary Report completed.
* All pre-defined testing activities has been completed to expected levels.
* All high priority bugs fixed, tested and passed.
* Test case granted “**PASS**” status.

**5.2.3 Suspension and Resumption Criteria**

|  |  |
| --- | --- |
| **Suspension Criteria** | **Resumption Criteria** |
| Test case will be suspended if it meet one or more of the criteria below. | Test case will be resumed if it meet one or more of the criteria below. |
| Version build is incorrect. | Version build corrected. |
| Testing software or hardware failure. | Testing software and hardware fixed and work properly. |
| Test environment is incorrect. | Test environment corrected. |
| Test case failed to achieve its minimum pass rate. | Bugs and defects fixed and approved by Quality Assurance Manager. |

**5.3 Testing Measurements and Metrics**

**5.3.1 Testing Measurements**

All testing process will be conducted and supervised by Quality Assurance Leads and coordinated by Quality Assurance Manager. It is important for the Quality Assurance Leads and Quality Control Leads to keep track of the following:

* Number of test cases passed
* Number of test cases failed
* Number of test cases not implemented during a particular test cycle.
* Number of test cases added or deleted
* Number of test cases re-executed
* Time taken for test cases execution

Business requirements are used as a guide to review the testing efficiency done.

**5.3.2 Testing Metrics**

All the data taken from the testing measurements are converted more useful information. The following are the Calculated Metrics are recommended to have:

* Percentage of complete test cases
* Percentage of defects corrected in a test cycle
* Percentage of re-discovered defects
* Percentage of test cases passed
* Percentage of test cases failed
* Percentage of test effectiveness
* Percentage of test cases suspended
* Percentage of test efficiency
* Percentage of failures
* Rate of defect discovery
* Rate of defect removal and cost

**Calculated Metrics**

Review efficiency is calculated to give an insight on quality of review and tests conducted. The higher the value, the more effective the review process implemented and defect spotting.

**Review efficiency** =

Cost Variance is calculated to view the company’s economic situation at the moment. It helps in analysing the project costs in reference with budgeted expenditure. It is extremely important because all developments cost money. If the result is positive, the company is experiencing losses and need to decide on how to cut cost.

**Cost Variance** =

Effort Variance is calculated to identify the effort spent in the project in terms of man-hours. It is directly linked to the Cost Variance and indicates the effectiveness of the allocation of resources.

**Effort Variance** =

Schedule Variance is calculated to measure the project completion of planned activities in reference of project schedule. It shows the effectiveness of the project scheduling and planning. Positive value indicates that the project is behind schedule while negative means the project is ahead of schedule.

**Schedule Variance** =

Cost of Quality reveals the total amount of effort spent on maintaining, refining and upgrading the software. Prevention Activities are time spent of planning, training and defect prevention. Verification and Validation are time spent of reviews, inspections and testing. Post Testing are effort which includes bug fixing, retesting and any correction effort that performed after testing occurred.

**Cost of Quality** =

Defect Occurrence can be calculated as the total number of defects compared to its size. The size of the project can be obtained from the functional point in the application.

**Defect Occurrence** =

Mean Time Between Failures is the average time between two critical system failures. It will give the user a time period to expect the next failure to occur.

**Mean Time Between Failures** =

Defect Removal Effectiveness is the total number of defects of a specific phase with reference of the total number of defects.

**Defect Removal Effectiveness** =

**5.4 Documentation**

**5.4.1 Test Plan Identifier (ID)**

**The ID for this test plan: TPN-v0.1a**

**Naming Convention Format for all related documents**

Example of an ID:

TC-MV-WK-UT-v0.1a-01

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Document Class ID | Feature Category | Feature  ID | Test  Levels | Version  Number | Assigned Group ID |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document Class ID** | | | | |
| To identify the document type, i.e. Test Strategy, Test Plan, Test Suite, Test Case. | | | | |
| **Document Type** | | | **Tags or IDs** | |
| Test Plan | | | TPN | |
| Test Suite | | | TS | |
| Test Case | | | TC | |
| **Feature Category** | | | | |
| To identify the category of a group of features, i.e. Movement, Mana, Health. | | | | |
| **Category** | | | **Tags or IDs** | |
| Movement | | | MV | |
| Combat | | | CMB | |
| Mana | | | MP | |
| Camera | | | CAM | |
| Stamina | | | STM | |
| Health | | | HP | |
| Mount | | | MNT | |
| Environment | | | ENV | |
| Model | | | MOD | |
| Levelling | | | LVL | |
| UI | | | UI | |
| Shop Mechanic | | | SHP | |
| Inventory | | | INV | |
| Shouts | | | SHT | |
| Quest | | | QST | |
| Character Creation & Customization | | | CCC | |
| UX | | | UX | |
| Audio | | | AUD | |
| Animation | | | ANM | |
| Miscellaneous | | | MSC | |
| **\*\*Regression Test** | | | RGT | |
| **Feature ID** | | | | |
| To identify the type of feature needed to be tested, i.e. Walk, Run, Crouch, Normal Attack | | | | |
| **Feature Category (ID)** | **Feature** | | | **Tag** |
| Movement (MV) | Walk | | | WK |
| Run | | | RUN |
| Crouch | | | CRH |
| Jump | | | JMP |
| Combat (CMB) | Physical Melee | | | MEL |
| Physical Ranged | | | RGD |
| Magic | | | MGC |
| Mana (MP) | Reduce | | | RDC |
| Replenish | | | RPN |
| Camera (CAM) | Camera movement | | | MV |
| Stamina (STM) | Reduce | | | RDC |
| Replenish | | | RPN |
| Health (HP) | Reduce | | | RDC |
| Replenish | | | RPN |
| Mount (MNT) | Mount | | | MNT |
| Unmount | | | UMT |
| Environment (ENV) | Terrain | | | TRN |
| Lighting | | | LGT |
| Shaders | | | SHD |
| Props | | | PRP |
| Structure | | | STC |
| Model (MOD) | Humanoid Character | | | HMN |
| Non-humanoid Character / Monster | | | MON |
| Leveling (LVL) | Experience Management | | | EXM |
| Gold Gain | | | GGN |
| UI (UI) | Menu UI | | | MUI |
| In-game UI | | | GUI |
| Shop Mechanic (SHP) | Buy | | | BUY |
| Sell | | | SEL |
| Inventory (INV) | Storage Management | | | STR |
| Equipping | | | EQP |
| Unequipping | | | UEQ |
| Enchantment | | | ECH |
| Shouts (SHT) | Shout Triggers | | | TRG |
| Quest (QST) | Quest Sequence | | | SEQ |
| Status Checking | | | STA |
| Character Creation & Customization (CCC) | Customization Stats Management | | | CSM |
| Model Adjustments | | | MDA |
| UX (UX) | Menu Visual Effects | | | MUX |
| In-game Visual Effects | | | GUX |
| Audio (AUD) | Background Music | | | BGM |
| Sounds Effects | | | SFX |
| Animation (ANM) | Animation Triggers | | | TRG |
| Animation Sequence | | | SEQ |
| Miscellaneous (MSC) |  | | |  |
| **Test Levels** | | | | |
| To identify the test levels of the document, i.e. Unit Test, Integration Test and System Test. | | | | |
| **Test Levels** | | **Tag** | | |
| Unit Test | | UT | | |
| Integration Test | | IT | | |
| System Test | | ST | | |

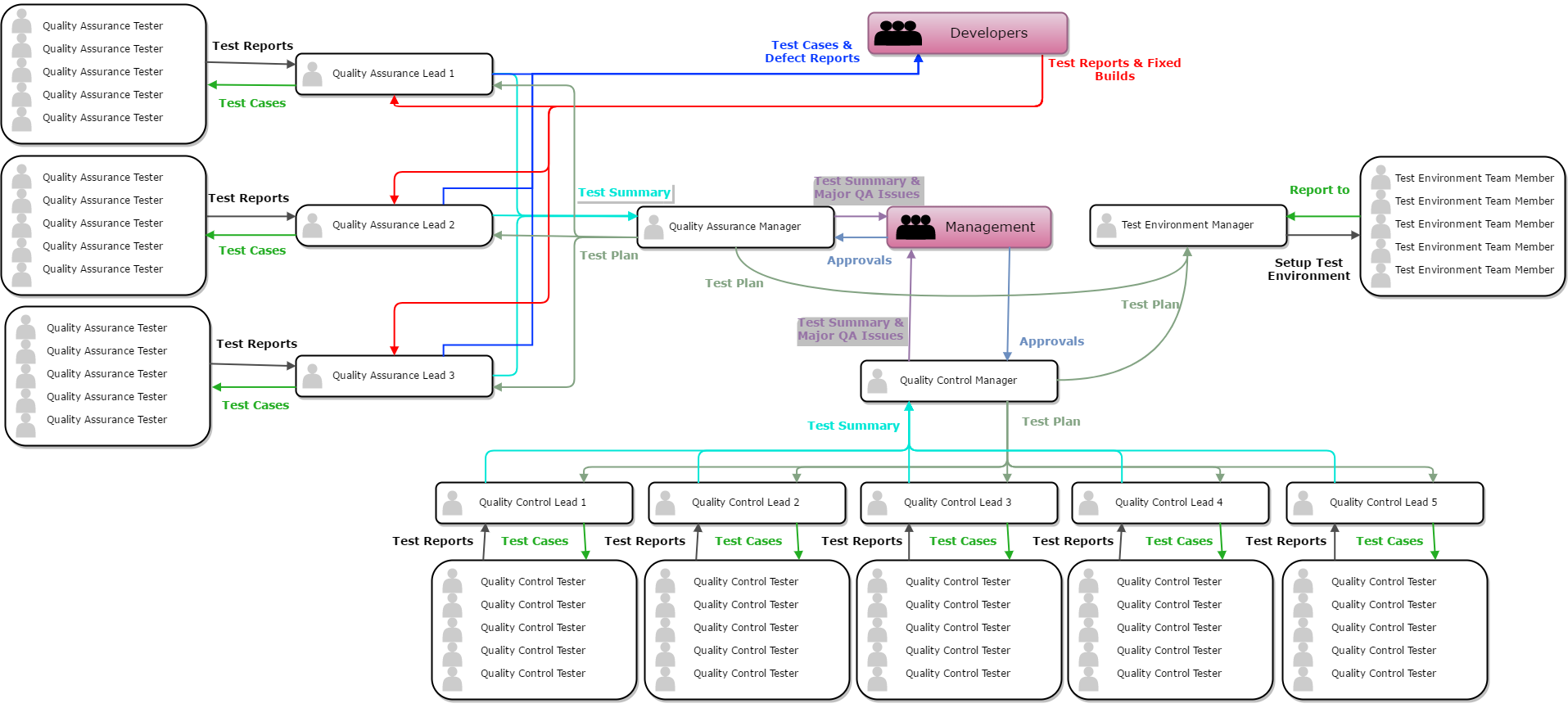
**5.4.3 Test Case**

Test Case is a document derived from a Test Plan. It is formulated to act as a guide the tester on its test inputs, execution procedures and expected results. A formal test case document will have the following contents:

* Test case ID
* Test title
* Test priority
* Test category
* Test level
* Test type
* Test execution date
* Name of tester
* Name of author
* Test case description
* Entry requirements
* Exit Requirements
* Test procedure
* Execution steps
* Required Data
* Expected result
* Test Result
* Status (Pass or Fail)
* Remarks

**5.5 Communication and Status Report**

Communication is extremely important in testing because the feedback generated from the tests could impact a game in all different aspects. So, communication must be done with the correct person-in-charge in order to maximise the efficiency of testing. The diagram below shows the correct communication pathways if communication is needed.

**5.5.1 Test Deliverables**

Correct Communication Pathways

Test deliverables are the reports filed after a test activity.

**5.5.1.1 Test Report**

Test Report is submitted after a test. It contains all the test results obtained from the executed test cases. It provides detailed descriptions and explanations of how the test results to determine whether the executed test cases passed the tests. A test report should provide the following content:

* Build release number
* Build release date
* Test case reference ID
* Test environment
* Test level & test types
* Total number of test cases
* Number of test cases conducted
* Number of test cases yet to be conducted
* Number of passed test cases
* Number of failed test cases
* Total number of bugs
* Detailed description of the bug
* Steps to reproduce the bug
* Bug severity & test scenario priority

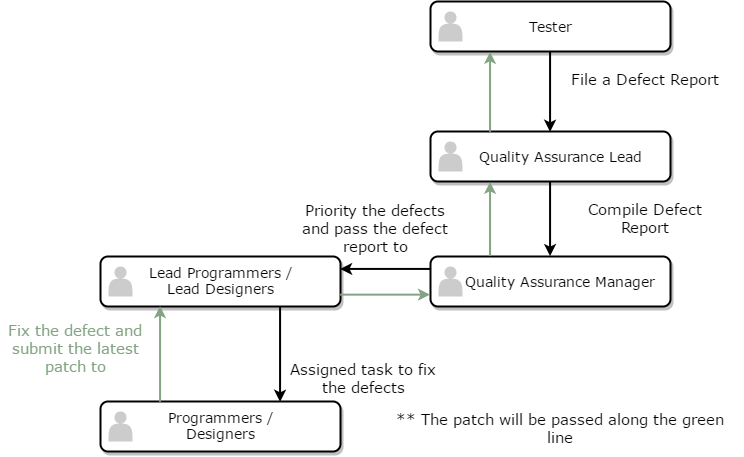
**5.5.1.2 Test Summary**

Test Summary is prepared after all the test reports on a specific build were collected. The purpose of test summary is to briefly explain various details and activities about the test activities performed. It also serves as a guide for the Quality Assurance Manager and Quality Control Manager to plan future tests. A test summary should provide the following content:

* Build release number
* Build release date
* Test case reference ID
* Test environment
* Test Levels & test types
* Total number of test cases
* Number of test cases conducted
* Number of test cases yet to be conducted
* Number of passed test cases
* Number of failed test cases
* Total number of bugs
* Bug severity & test scenario priority
* Requirement traceability matrix

**5.5.1.3 Defects Report and Tracking**

The flowchart below shows how defects reporting, tracking, fixing and patching works.



Bugs and defects found during testing phases are to be reported by filling up the Bug Report Form below by the tester. The form is then submit to the tester’s Lead for compilation. After compilation, the Leads will then report the defects to the Quality Assurance Manager or Quality Control Manager.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bug Report Form** | | | | | |
| **Tester Name** | |  | | | |
| **Date** | |  | | | |
| **Build Version** | |  | | | |
| **No.** | **Bug Severity** | | **Bug Description** | **Bug Occurrence Rate** | **Comments** |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |
|  |  | |  |  |  |

**5.5.2 Test Progress Report**

Test Progress Report is submitted either daily, weekly or monthly. It reports information about the current testing progress for the Leads and Managers to plan their future schedule and tests. The following is the Test Progress Report.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Progress Report** | | | | | | | |
| **Tester Name** | |  | | | | | |
| **Test Case**  **ID** | **Date** | | **Pass / Fail** | **Defect Description** | **Defect**  **Severity** | **Bugs Status** | **Comments** |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |
|  |  | |  |  |  |  |  |

**5.6 Risk and Mitigation**

The table below shows the possible risks that can happened during development and testing and the possible solution to mitigate the current contingencies.

|  |  |
| --- | --- |
| **Risk** | **Mitigation** |
| Testers unable to test their assigned test cases. | * Find other testers that are available. |
| If hardware and software breaks down. | * Contact the IT department and ask for their assistance. |
| If test environment setup does not match the test environment stated in the test case. | * Contact and ask for Test Environment Manager’s assistance. |
| If a member of Test Environment Team is unavailable to work, resigned or permanently left the job. | * Test Environment Manager should start preparing the test environment earlier. * Hire new Test Environment Team member. |
| If a Quality Assurance Lead is unavailable to supervise the test. | * Quality Assurance Manager can send other teams’ Quality Assurance Lead to supervise the test. * Quality Assurance Manager supervises the test himself / herself. |
| If a Quality Control Lead is unavailable to supervise the test. | * Quality Control Manager can send other teams’ Quality Control Lead to supervise the test. * Quality Control Manager supervises the test himself / herself. |
| If a Quality Assurance Tester resigned or permanently left the job. | * Quality Assurance Manager can to reschedule the tester groupings. * Quality Assurance Manager shifts some of the work to other tester groups. * Hire new Quality Assurance Tester. |
| If a Quality Assurance Lead resigned or permanently left the job. | * Quality Assurance Manager has to reschedule the tester groupings. * Quality Assurance Manager temporary takes the Quality Assurance Lead position and coordinates the tester group. * Hire new Quality Assurance Lead. |
| If a Quality Control tester resigned or permanently left the job. | * Quality Control Manager can to reschedule the tester groupings. * Quality Control Manager shift some of the work to other tester groups. * Hire new Quality Control Tester. |
| If a Quality Control Lead resigned or permanently left the job. | * Quality Control Manager has to reschedule the tester groupings. * Quality Control Manager temporary takes the Quality Control Lead position and coordinates the tester group. * Hire new Quality Assurance Lead. |

**5.7 Change Control Management**

**5.7.1 Document Control Form**

Any iterations of the document must be logged with the Document Control Form to keep track of any changes happened. This document is essential because if unwanted changes happened the Quality Assurance and Quality Control teams can revert back to the previous versions. The following is a template of the Document Control Form.

**Document Control Form**

**Document Detail**

|  |  |
| --- | --- |
| **Title** |  |
| **Version** |  |
| **Date** |  |
| **Electronic File Name** |  |
| **Electronic File Location** |  |
| **Authors** |  |
| **Contributors** |  |

**Document Change Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Issue Date** | **Version** | **Details** | **Authors** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Referenced Documentation**

|  |  |  |
| --- | --- | --- |
| **No.** | **Document Name** | **Document File Location** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**5.7.2 Change Request Form**

If changes in the system is needed to be performed, testers are required to fill up Change Request Form. After filling the form, the form is then send to Quality Assurance Leads for review and then pass to Quality Assurance Manager and Management for approval. After the form, is passed back to the developers for change development. The following is a template of the Change Request Form.

|  |  |
| --- | --- |
| **Change Request Form** | |
| **Project** |  |
| **Change Requestor** |  |
| **Date** |  |
| **Change No.** |  |
| **Change Category** | *Check when applied:*  Correction Cost Scope  Requirements Deliverables Testing  Resources |
| **Affected Items** | *Check when applied:*  Correction Prevention Defect Fix  Updates Others |
| **Description of the Change Requested** |  |
| **Reason of Change** |  |
| **Alternatives** |  |
| **Technical Changes Needed** |  |
| **Risks to be Considered** |  |
| **Estimated Resources and Costs Needed** |  |
| **Status** | Approve Reject Defer |
| **Justification of Status Given** |  |
| **Approvals** | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_  ( ) ( )  Quality Assurance Quality Control  Manager Manager  Date: Date:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_  ( ) ( )  Development Management  Project Manager Date:  Date: | |

**6. Requirement Traceability Matrix**

Requirement Traceability Matrix is tabulated to ensure the development meets the business or development requirements set by the management or the developers.

**The objectives of the project is to:**

* Create a unique and more diverse game world than Oblivion
* Reflect the socioeconomic background of NPCs
* More immersive world with more realistic elements
* Emphasize more on player’s discovery and exploration.

Business / Development Requirements, R(x), x = Requirement Index

x = 1. Immersive world with more realistic elements

x = 2. Emphasize more on player exploration

x = 3. Difference in socioeconomic status

x = 4. Smoother character movements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Feature**  **Category** | **Feature** | **Requirements** | | | |
| **R1** | **R2** | **R3** | **R4** |
| Movement | Walk |  | X |  | X |
| Run |  | X |  | X |
| Crouch |  | X |  | X |
| Mana | Reduce | X |  |  |  |
| Replenish | X |  |  |  |
| Health | Reduce | X |  |  |  |
| Replenish | X |  |  |  |
| Environment | Terrain | X | X |  |  |
| Lighting | X |  |  |  |
| Props | X | X | X |  |
| Model | Humanoid Character |  |  | X | X |
| Non-humanoid Character |  |  | X | X |
| Levelling | Experience Management |  |  |  |  |
| UI | Menu UI |  |  |  |  |
| In-game UI |  | X |  |  |
| Shop Mechanic | Buy | X |  | X |  |
| Sell | X |  | X |  |
| Inventory | Sorting | X |  |  |  |
| Storage Management | X |  |  |  |
| Equipment | Equipping |  |  |  | X |
| Unequip |  |  |  | X |
| Enchantment | X | X |  |  |
| Quest | Quest Sequence | X | X |  |  |
| Status Checking | X |  |  |  |
| Character Creation & Customization | Customization Stat Management |  |  | X |  |
| Model Adjustments |  |  |  |  |
| UX | Menu Visual Effects |  |  |  |  |
| In-game Visual Effects | X |  |  |  |
| Audio | Background Music | X | X |  |  |
| Sounds Effects | X | X |  |  |
| Animation | Animation Triggers |  |  |  | X |
| Animation Sequence |  |  |  | X |
| Miscellaneous |  | X | X | X | X |

**7. Approval**

Please leave signature in the blank space below if you approve the test strategy. Signing on the blank space will approve this document with immediate effect. Approvers are liable for any sort of punishment and are responsible for the loss, damage or any unexpected negative impact of the application.

|  |  |  |
| --- | --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| ( ) |  | ( ) |
| Quality Assurance Manager |  | Quality Control  Manager |
| Date: |  | Date: |
|  |  |  |
|  |  |  |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
|  | ( ) |  |
|  | Project Manager |  |
|  | Date: |  |