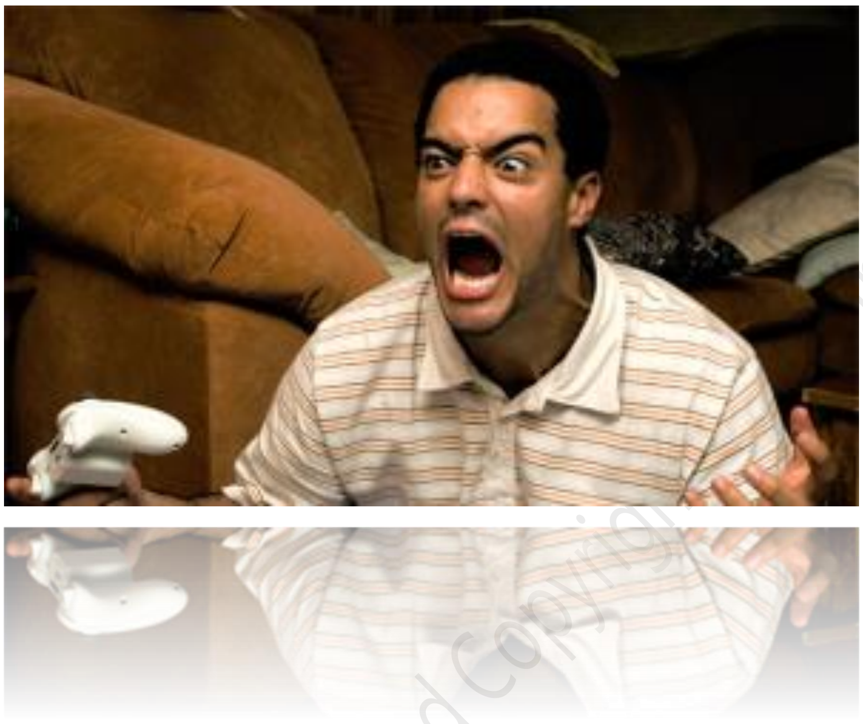


Game Testing

Aurthor

Kitman Yiu

Project Architecture 2

Teacher: BARKAT ULLAH

CONTENT

INTRODUCTION

`In this article we will talk about what is Game testing, and why game testing is important, after that we will introduce to the process of how software has been tested and what types and level of testing a software , at the end I will present a document that what game industry can follow.

GAME TESTING

INTRODUCTION

In game industry, to allow the company to maximum their profit. It is important to reaches the customers' requirement, and to reaches what the customer need to test is one of the most important step, in game industry one of the keyword we use is "**game testing**". Game testing is a subset of game development, the connection between game development and game testing is showed as by the following image:

WHAT IS GAME TESTING

Is a software testing process for quality control of video games? The primary goal of game testing is the discovery and documentation of software defects (aka bugs)

HOW TO DO GAME TESTING?

Over the years a number of types of definition have been invented to allow for the control of testing.so IEEE developed the 1926 Standard and 829 to provide a common set instruction and documents to test any type of software testing(not only for game industry).

WHEN WILL THE GAME TESTING BE START AND END?

The testing will start as soon as the first code is written and increase as the game progresses towards completion. The main QA team will monitor the game from its first submission to the QA until as late as post-production. For big companies. In the early stage the testing team is small and focuses on daily feedback for new code.

WHO WILL DO GAME TESTING?

To able to do game testing .Small development will not hire QA staff and choose the do play testing by itself or the friends that they trust. However. Large companies may employ QA teams full-time.

WHERE IS THE GAME TESTING?

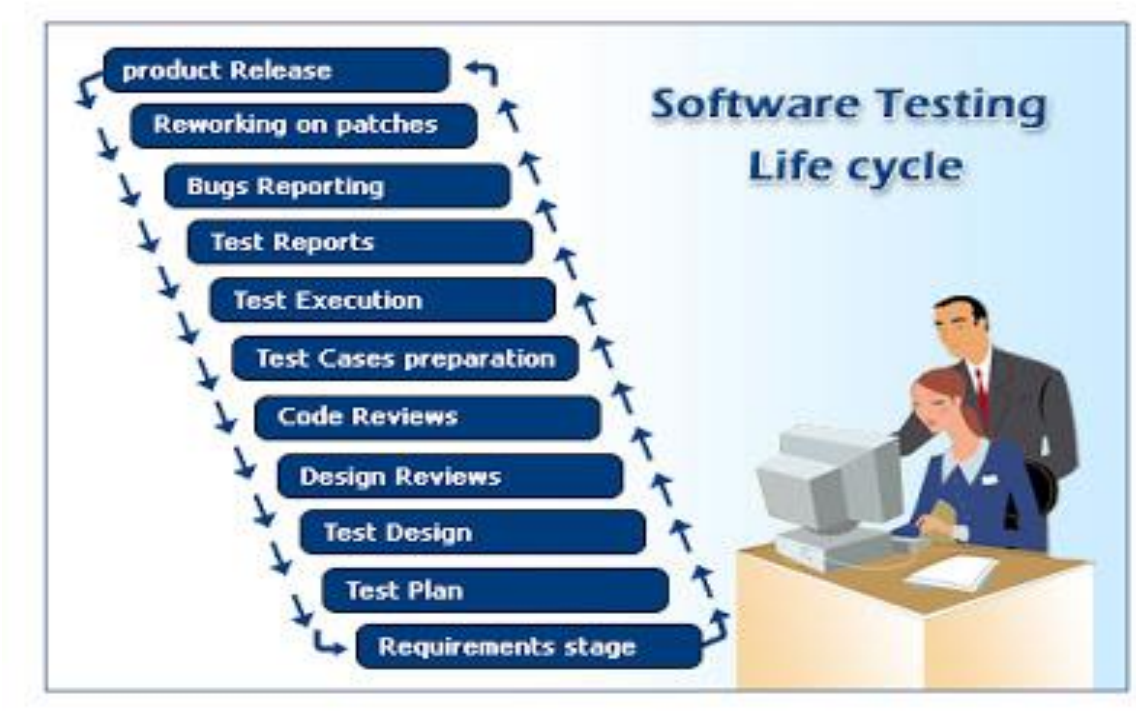
In small or indie companies the play test it usually can be anywhere, even when having lunch.

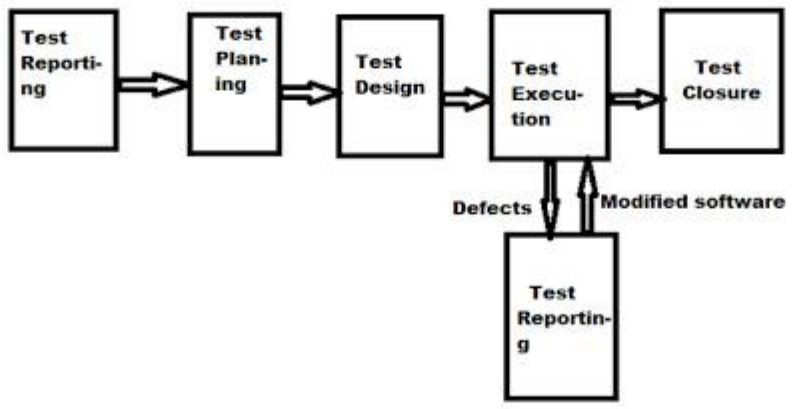
HOW MUCH THE GAME TESTING DOES HELPS TO MAXIMUM THEIR PROFIT?

There is no

SOFTWARE TESTING LIFE CYCLE

According to http://ashishqa.blogspot.com.au/2013/06/software-testing-life-cycle.html





In General, System testing process or STLC starts with test initiation or test commencement. In this phase, project manager or Test Manager selects the reasonable approaches which are followed by Test Engineers. He prepares a document called TEST STRATEGY DOCUMENT in IEEE 829 Format.

IEEE 9126 (ISO)

INTROUCTION

(According to SQA) Before we talk about the TEST STRATEGY DOCUMENT (IEEE 829), we first need to talk about IEEE 9126 because over the years a number of types of definition have been invented to allow for the control of testing. They apply to software testing of all kinds from component testing through to release testing. Every organization develops these documents themselves and gives them different names, and in some cases confuses their purpose. To provide a common set of standardized the IEEE developed the 1926 Standard for Software Test for any type of software testing.

6 MAIN QUALITY CHARACTERISTICS

The ISO 9126-1 software quality model identifies 6 main quality characteristics, namely:

- Functionality
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

Functionality
Functionality is the essential purpose of any product or service. For certain items this is relatively easy to define.

Reliability
Once a software system is functioning, as specified, and delivered the reliability characteristic defines the capability of the system to maintain its service provision under defined conditions for defined periods of time. One aspect of this characteristic is fault tolerance that is the ability of a system to withstand component failure. For example if the network goes down for 20 seconds then comes back the system should be able to recover and continue functioning.

Usability
Usability only exists with regard to functionality and refers to the ease of use for a given function. For example a function of an ATM machine is to dispense cash as requested. Placing common amounts on the screen for selection, i.e. $20.00, $40.00, $100.00 etc, does not impact the function of the ATM but addresses the Usability of the function. The ability to learn how to use a system (learnability) is also a major subcharacteristic of usability.

Efficiency
This characteristic is concerned with the system resources used when providing the required functionality. The amount of disk space, memory, network etc. provides a good indication of this

characteristic. As with a number of these characteristics, there are overlaps. For example the usability of a system is influenced by the system's Performance, in that if a system takes 3 hours to respond the system would not be easy to use although the essential issue is a performance or efficiency characteristic.

Maintainability
The ability to identify and fix a fault within a software component is what the maintainability characteristic addresses. In other software quality models this characteristic is referenced as supportability. Maintainability is impacted by code readability or complexity as well as modularization. Anything that helps with identifying the cause of a fault and then fixing the fault is the concern of maintainability. Also the ability to verify (or test) a system, i.e. testability, is one of the sub characteristics of maintainability.

Portability
This characteristic refers to how well the software can adopt to changes in its environment or with its requirements. The sub characteristics of this characteristic include adaptability. Object oriented design and implementation practices can contribute to the extent to which this characteristic is present in a given system.

IEEE 829 (ISO)

INTRODUCTION

There was a IEEE 829 – 1998 version which isn't the newest the newest the newest version is IEEE 2008.If a properly balanced test plan is created then a project stands a chance of delivering a system that will meet the user's needs.

TYPES OF DOCUMENT

(According to SlideSharing) The IEEE 829 provides a common set of standardized documents the IEEE developed the 829 Standard for Software Test Documentation for any type of software testing, including User Acceptance Testing. A test plan is a specific version of a project plan with clauses that meet these requirements. The main characteristics of a project plan are described in the article Basics of a Project Plan. The international standard IEEE Std 829-1998 gives advice on the various types of test documentation required for testing including test plans, and details of these are in the IEEE 829 article. The test plan section of the standard defines 16 clauses. This article gives guidance on how the IEEE 829 standard maps against the requirements of a project test plan.

There are eight document types in the IEEE 829 standard, which can be used in three distinct phases of software testing:

- *Test Plan Document(ISO 829)*

Plan how the testing will proceed. An example is showed in the next parpragh.

- *Test Design Specification:*

Decide what needs to be tested.

- *Test Case Specification:*

Create the tests to be run. An example is showed by the following link

- *Test Procedure:*

Describe how the tests are run. An example is showed by below:

- *Test Item Transmittal Report:*

Specify the items released for testing. An example is showed by below

- *Test Log:*

Record the details of tests in time order. An example is showed below

- *Test Incident Report(ISO 829-1998)*

Record details of events that need to be investigated. An example is showed below

- *Test Summary Report:*

Summaries and evaluate tests. An example is showed below

The following website is an example template of each document

EXAMPLES

Test Plan Document(ISO 829)

An example is showed in the next parpragh.

Test Procedure:

http://www.dot.state.fl.us/trafficoperations/ITS/Projects_Deploy/SEMP/ApxK.pdf

Test Item Transmittal Report

http://qtp.blogspot.com.au/2009/06/test-item-transmittal-report.html3

Test Log:

https://www.google.com.au/url?sa=t&rct=j&q=&esrc=s&source=web&cd=7&ved=0CDgQFjAG&url=http%3A%2F%2Fwww.bestpractices.osi.ca.gov%2Fsysacq%2Fdocuments%2FTest%2520Log%2520Template.doc&ei=3wBCVJ-MJeS7mQW6lYLwCQ&usg=AFQjCNH6NveshZ05vMsq0VpSLOzwKDUHqA&sig2=Vmujul2xIjbCSj2q88swXA&bvm=bv.77648437,d.dGY&cad=rja

Test Incident Report(ISO 829-1998)

https://courses.cs.ut.ee/MTAT.03.159/2014_spring/uploads/Main/SWT_test-incident-report-template.pdf

Test Case Specification

https://docs.google.com/document/d/1qH_yp0a4epaRwtuI5WXp_N3MxQ56sWBDvROcJ08bDMo/preview

Test Summary Report:

https://www.google.com.au/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=0CCUQFjAB&url=http%3A%2F%2Fwww.cambridge.org%2Fus%2Fdownload_file%2F202219%2F&ei=OQFCVI3kI6LMmwXHt4CYDw&usg=AFQjCNEzdeZHBV8qXa4tEUk4-Zntio92_g&sig2=qY9n0IaJCilG5FczQptfAg&bvm=bv.77648437,d.dGY&cad=rja

(According to ColeyConsulting). The 16 clauses of the IEEE 829 Test Plan Document standard are:

1. Test plan identifier.
2. Introduction.
3. Test items.
4. Features to be tested.
5. Features not to be tested.
6. Approach.
7. Item pass/fail criteria.
8. Suspension criteria and resumption requirements.
9. Test deliverables.
10. Testing tasks.
11. Environmental needs.
12. Responsibilities.
13. Staffing and training needs.
14. Schedule.
15. Risks and contingencies.
16. Approvals.

1. Test Plan Identifier: This is a unique name or code by which the plan can be identified in the project's documentation including its version.
2. Introduction: A high level view of the testing standard required, including what type of testing it is.
3. Test Items: The items of software, hardware, and combinations of these that will be tested.
4. Features to Be Tested: The parts of the software specification to be tested.
5. Features Not to Be Tested: The parts of the software specification to be excluded from testing.
6. Approach: The details of how the testing process will be followed including Entry Criteria and Exit Criteria. (White box testing and black box testing)
7. Item Pass/Fail Criteria: Defines the pass and failure criteria for an item being tested.
8. Suspension Criteria and Resumption Requirements: This is a particular risk clause to define under what circumstances testing would stop and restart.
9. Test Deliverables: Which test documents and other deliverables will be produced. The associated article on test documentation gives details of the IEEE 829

documentation.
10. Testing Tasks: The tasks themselves, their dependencies, the elapsed time they will take, and the resource required.
11. Environmental Needs: What is needed in the way of testing software, hardware, offices etc.
12. Responsibilities: Who has responsibility for delivering the various parts of the plan
13. Staffing and Training Needs: The people and skills needed to deliver the plan.
14. Schedule: When the tasks will take place.
Often these two clauses refer to an appendix or another document that contains the detail.
15. Risks and Contingencies: This defines all other risk events, their likelihood, impact and counter measures to overcome them.
16. Approvals: The signatures of the various stakeholders in the plan, to show they agree in advance with what it says.

TESTING LEVEL

- *Unit Testing*
- *Component Testing*
- *Integration Testing*
- *System Testing*
- *Acceptance Testing*
- *Alpha Testing*
- *Beta Testing*

TESTING TYPES

FUNCTIONAL TESTING

- *Installation*
- *Development*
- *Usability*
- *Sanity*
- *Smoke*

- *Regression*
- *Destructive*
- *Recovery*
- *Automated*
- *User Acceptance*

NON FUNCTIONAL TESTING

- *Compatibility*
- *Performance*
- *Security*
- *Accessibility*
- *Localization*

TESTING METHODS

(According to youtube) There are more than 150 types of testing we will introduce some common and basic testing methods

(According to Buzzle) The common testing type are
Unit Testing - The first to be carried out is the unit test. As the name suggests, this method tests at the object level. Individual software components are tested for any errors. Accurate knowledge of the program is needed for this test, as each module is checked. Thus, this testing is done by the programmers and not the testers. Test codes are created to check if the software behaves as it is intended to.

Integration Testing - Individual modules that are already subjected to unit testing are integrated with one another, and are tested for faults. Such a type of testing highlights interfacing errors. A 'top-down' approach of integration testing follows the architectural structure of the system. Another approach taken is the 'bottom-up' approach, which is conducted from the bottom of the control flow.

System Testing - In this testing, the entire system is tested for errors and bugs. This test is carried out by interfacing hardware and software components of the entire system, and then testing it. This testing is listed under the black-box testing method, where the software is checked for user-expected working conditions.

Acceptance Testing (Beta test) - This is the last test that is conducted before the software is handed over to the client. It is carried out to ensure that the software that has been developed meets all customer requirements. There are two types of acceptance testing - one that is carried out by the members of the development team, known as internal acceptance testing (Alpha testing), and the other that is carried out by the customer, known as external acceptance testing. If the testing is carried by the intended customers, it is termed as customer acceptance testing. In case the test is performed by the end users of the software, it is known as user acceptance testing (Beta testing).

The basic of testing type are

Black-box Testing - Black-box testing is carried out without any knowledge of the internal working of the system. The tester will stimulate the software to user environment by providing different inputs and testing the generated outputs. This test is also known as closed-box testing or functional testing.

White-box Testing - White-box testing, unlike the black-box one, takes into account the internal functioning and logic of the code. To carry out this test, the tester should have knowledge of the code, so as to find out the exact part of the code that is having errors. This test is also known as open-box testing or glass testing.

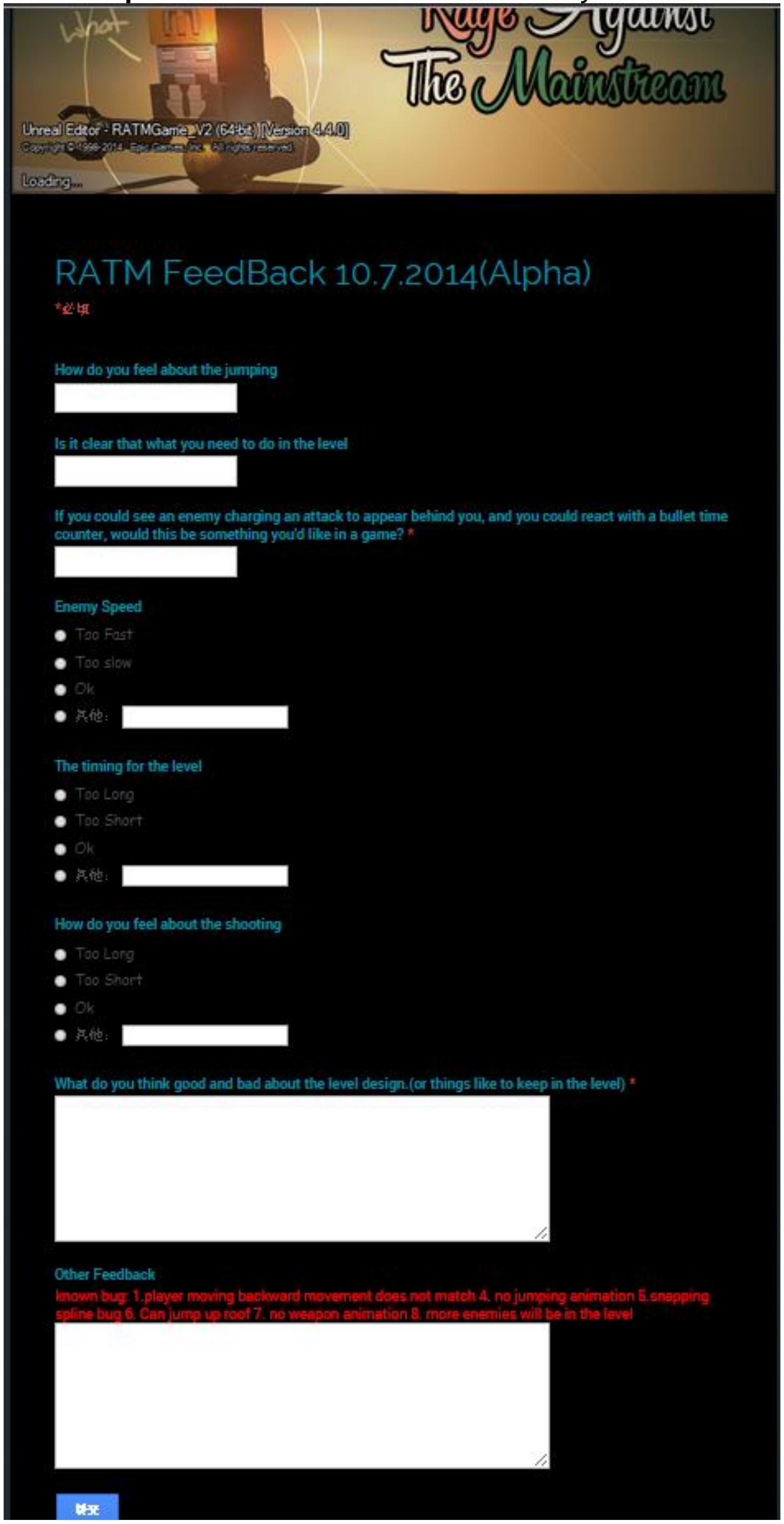
Gray-box Testing - The testing where part knowledge of the code is necessary to carry out the test is called gray-box testing. This testing is done by referring to system documents and data flow diagrams. The testing is conducted by the end users, or users who pose as end users.

Because the time and budget limitation, I would recommend for small/indie companies, I would recommend at least

Each week have a 1 document that include what things do you want to test, and a bug list that is already known and show the tester so when giving other feedbacks does not give any information that’s already know. If this test have been more than 80% of person give a feedback is positive then the task will pass else the question will be go to next weeks until the task has been proved.

- Level Design Questions
- Charthers Control Questions
- User Interface Questions
- AI/Enemy Questions

An example is showed below Online Survey


What
Rage Against
The Mainstream
Unreal Editor - RATMGame_V2 (64-bit) [Version 4.4.0]
Copyright © 1998-2014 Epic Games, Inc. All rights reserved.
Loading...

RATM FeedBack 10.7.2014(Alpha)

*必填

How do you feel about the jumping

Is it clear that what you need to do in the level

If you could see an enemy charging an attack to appear behind you, and you could react with a bullet time counter, would this be something you'd like in a game? *

Enemy Speed

- Too Fast
- Too slow
- Ok
- 其他：

The timing for the level

- Too Long
- Too Short
- Ok
- 其他：

How do you feel about the shooting

- Too Long
- Too Short
- Ok
- 其他：

What do you think good and bad about the level design.(or things like to keep in the level) *

Other Feedback

known bug: 1.player moving backward movement does not match 4. no jumping animation 5.snapping spline bug 6. Can jump up roof 7. no weapon animation 8. more enemies will be in the level

Document Example:

However for a big company I would recommend big companies:
Follow the iso standard the benefits are listed below:

- Record of what was tested
- Can help have strategies and tactics to help you and you your team to test quicker, with more accuracy.
- To check the reliability of the software.
- To be ensured that the software does not contain any bug which can become a reason for failure.
- To check the software was made according to its specification.
- To check that the software meets its requirements.
- To check that users are capable of using the software.
- To check software works with other software and hardware it needs to work with

CONTENT

SlideSharing.2013. *Software testing methods, levels and types.* [ONLINE] Available at: http://www.slideshare.net/confiz/software-testing-methods-levels-and-types [Accessed 8 October 14].

ColeyConsulting. *IEEE 829 Test Plan*. [ONLINE] Available at: http://www.coleyconsulting.co.uk/testplan.htm. [Accessed 8 October 14].

SQA. *ISO 9126 Software Quality Characteristics*. [ONLINE] Available at: http://www.sqa.net/iso9126.html [Accessed 8 October 14].

wiki. 2014. *Game Testing*. [ONLINE] Available at: http://en.wikipedia.org/wiki/Game_testing. [Accessed 8 October 14].

youtube. 2008. *Test Plan Template for Software Testing Projects*. [ONLINE] Available at:https://www.youtube.com/watch?v=ddiIm3TsBJg. [Accessed 19 October 14].

Software Testing Methodologies. 2014. *Test Plan Template for Software Testing Projects*. [ONLINE] Available at: http://www.buzzle.com/articles/software-testing-methodologies.html. [Accessed 19 October 14].