Research paper on computerized game development

Business, Customers

Introduction

Software development is also known as software design and involves the development of a software product. Software development can also refer to computer programming which involves writing and maintaining source codes of a given computer program. The process of software development is very wide and includes researching on a particular kind of software, making new developments or modifying an already existing software, prototyping, requirements engineering, maintenance and carrying any other activity that can result into the formation of a software product. Software development is usually done for a variety of purposes. In each of the purposes that may trigger the development of software, customer satisfaction is usually given a priority. Software developers often develop software that can meet the particular/customized needs of their clients. Software can also be developed in order to be able to accomplish some given kind of task. There is need to have a better quality software development process so as to have a systematic approach in the process of software development (Berenbach et al., 2009). This helps in ensuring that high quality software is produced. It also makes it possible for the maintenance team to identify a code that needs to be corrected incase the software experiences some technicalities. It is therefore necessary for any software developer to follow the stated rules when handling software development so as to have a unified approach acceptable to all. In this study, we are going to consider the development of a computerized game. This software can be used for playing computer games in different platforms. The process of developing a computer game involves a variety of procedures which must be adhered to in order to

produce a quality game that can serve the purposes it is meant for. The modern computer games are as a result of some complex products that use innovative research methods of graphics calculations, network interaction and also involve the physics of objects. Computer games can be developed in a variety of ways depending on the targeted platform. When writing the computer game, several programming languages and technologies can be used in the process. Some of the most commonly used software tools for writing computer games include flash, java and C+.

Software requirements

Software requirements are the basis of all the applications in software development. The requirements focuses on what the software being developed must do. If this phase is not correct, then companies will not end up with the software that they require for performing a particular job. In requirements engineering, both the software developer and the customer takes an active role in the design. The software requirements are elicited from the customers' specifications on what they expect from the software and the activities that they want the software to perform for them. Once the entire requirements have been gathered, a requirements document is then prepared. This document is prepared in a language that the customers can understand. The requirements document actually forms the contact between the software developers and the customers. The developers have to make software that meets the specifications indicated in the requirements document. It is essentially not possible for the software developers to make software that will meet all the requirements stated by the customer.

Therefore a strategy called Win-win model is used whereby some compromises have to be made. The developers therefore develop software that meets most if not all of the requirements stated in the document.

There are several ways of stating requirements. Formal requirements documents include those ones dealing with the functional and non-functional requirements. The functional requirements will describe how the developed system will interface with the environment. This essentially includes; the services that the system will provide to its users and how the system reacts to inputs from users. The non-functional requirements on the other hand describe the restrictions on the system. The restrictions which are included here involve time constraints, privacy policy and the security of the system.

In the requirements phase, Use cases and User stories can be used for documentation. The Use Cases shows the interactions existing between the users and the tasks that are to be performed by the system. User stories on the other hand are the short descriptions of tasks which are written by the customers. The user stories are useful in the process of defining the development priorities.

The requirements must be determined and then agreed on by different stakeholders in the process of development including the customers, suppliers and the developers of the software. In the process of stating the software requirements, the following needs to be taken into consideration:

i. Requirements elicitation

This is the first procedure that is conducted before requirements are analyzed. It is the process used for gathering the requirements.

ii. Requirements analysis

The requirements analysis phase will act as a bridge to the gap between the system level requirements and the software design. The figure below shows the role played by requirements analysis in requirements engineering.

iii. Requirements specification

This phase provides the developer and the customer with a means of determining and accessing the quality of the software once the development has been completed.

iv. Requirements validation

The software developers need to understand the different types and levels of requirements so as to have a good requirements engineering job done. They need to understand what is required for the successful development of the system so as to allocate adequate resources for the process and ensure that all the tasks are completed in a timely manner. In order to perform requirements engineering well, it is necessary to exercise an interdisciplinary approach which will consider all the needs of multiple stakeholder groups.

Modeling

Models helps in giving a glimpse of how the software will function and also give a representation of what is expected from the software. The models which are created during requirements analysis have several functions. They include:

Helping the analyst to understand the information being presented

It forms the focal point for carrying out a review in the system

It provides the foundation for the design and helps the analyst to determine

whether a given development is viable/possible

There are several models that can be used in software creation. They include:

i. Data models

Data models are abstract models which are used to document and organize data. The data models help in initiating communication between the team members and the application being developed. The main role of the data model is to support the development of the information system. The process of constructing data models involves stating all the requirements of the system and putting the corresponding codes for different functionalities of the system. The codes are then connected together and then executed in order to check their functionalities and viabilities. In the computer game development, all the user interactions are stated and then possible outputs also outlined. Once they have all been outlined, the developer codes them using a low level language into a simulator to determine how the system is likely to function. This then becomes the main building block for the computer game development.

ii. Functional model

These are created so as to gain a better understanding of the actual entity which is going to be built. The functional model must be able to represent the information that the software is supposed to transform. When creating the functional models for game development the developer needs to focus on problem specific functions. A functional detail is provided over a series of iterations until a thorough representation of the final product is represented. In the game development, several iterations are conducted using different

types of inputs. This process is repeated until all the possible inputs have been considered and that the desired output achieved. This can then be transferred to the final project which is being produced.

iii. Behavioral Models

Most software being designed responds to different events from the external environment. The stimulus that propagates the response forms the basis for making the behavioral models. A computer program will always exist in some externally observable mode of behavior which can only be changed when a given event occurs. A computer will remain in the standby state until the user inputs some instructions for the computer to behave otherwise. When a person inputs some instructions, there are several activities that take place in the computer thus instructing it to depict some kind of behavior. When there is an external stimulus to a system/software in the standby stay, an interrupt is created. Depending on the effect of the interrupt, the developer can then modify it so that it suits his/her needs in the process of software development. A behavioral model is therefore used to create a representation of the software states and the events that can be used to propagate a change in state of the software.

Designing data, architecture, interface and code

The process of designing the software is a complex process that must be done sequentially in order to ensure that none of the components are left unattended to. The process of designing the software was divided into three different phases after the requirements had been gathered.

The phases included making the models, coding the program and

implementation. After the software had been implemented, it was then tested to confirm if it met the specifications that were stated before commencing the project.

The user inputs were designed. The project being worked on is a racing computer program. In the user inputs section, three categories were considered namely; selection (car selection, track selection and nature of competition), racing and timing.

a) Selection

Specific cars had specific specifications which included a specified transmission, maximum speed, fuel consumption, braking and tracking control. The track selection involved a choice of tracks and obstacles that were to be met on the way. This selection also includes whether to race with cars or be a single competitor on the track. The last selection was the nature of race (whether competition or practice). All these were coded using java.

b) Racing

In this section there were comtrols that were used. Each control was assigned to a specific key. Whenever the key is pressed, it creates some interrupt which triggers an action depending on the coding done to the key. The up arrow key is assigned to acceleration, right key – turning right, left key – turning left and down key – braking. Each of the keys called for a procedure which was programmed in java. Whenever a procedure was called, an action was initiated which to the user appears as a racing car.

c) Timing

Once the enter key is pressed, a procedure is called. The procedure called int_time procedure is used for starting the clock. The clock is to remain

running until the ESC key is pressed or the finishing line crossed. The ESC key also calls another procedure called int_stoptimer. Once the race is completed the time is recorded.

The interfaces that is used for the program is keyboard. A user presses any of the designated keys which initiate a procedure depending on the coding done. The procedures then are executed and the right action performed.

Software Testing

Once the software has been designed, it is therefore necessary for the development team to carry out the testing procedure. Software testing is very essential as it helps in determining the viability of the software and ensuring that the software is performing the function it was intended to. There are several ways of performing the testing procedure. In the case of the game development, several sets of inputs were used so as to determine the outputs. If the outputs are undesired, then the developers have to go back to the software codes to determine the place that might be having a hitch and make the necessary correction.

Testing the software involved trying to make a car rotate. However the car could not rotate but the front part of the car remained facing the finishing line. This was done intentionally so that when a person is racing the car, he/she doesn’t move in the opposite direction. Feeding undesired input like pressing non-designated keys did not result into any output. Once the software has been tested and all the functionalities achieved, it can then be delivered to the customers for use.

Software engineering metrics

The software engineering metrics are used to provide some quantitative basis for the development and validation of different models which are used in the software development process. The metrics are used for improving the quality and productivity of the software being developed. In our case we used the following software metrics to ensure the quality of the software produced.

i. Defects per thousand lines of code

When using the defects per thousand lines of code some defects were discovered especially in giving the specifications for the car types and acceleration. Some of the cars could not accelerate as expected. When the codes were accessed, more modifications were made so as to enable the car increase their acceleration speed whenever they hit an obstacle or stopped abruptly. This was discovered during the testing phase and once the modifications were done, most cars were able to race perfectly well. This ensured that we produced a quality work that could meet customers' satisfaction to the latter.

ii. Cost performance index

This metric took into consideration the number of developers working on the project, the time the project was expected to be completed and the cost of the project. In the case of game development, only five developers were working on the program and the project was to last for a period of two months. Once the project was completed, it was to retail at $1000. We marked a low price so as to enable us get more customers so as to recover the initial cost of producing the software. Considering the amount of money

that was expected to be obtained from the sale of the software and the efforts put into the project, we can say that it was viable. We engaged dedicated and professional staffs who were able to deliver quality work hence ensuring a value for the money put into the project.

Software maintenance techniques

There are several maintenance techniques that were used in the process of carrying out the development. We used the following techniques for maintenance purposes.

i. Program comprehension

We provide a concise and precise documentation for the project. This was meant at ensuring that programmers could easily comprehend the codes used thus making it easy for them to trace whenever a problem occurred and then make the necessary adjustment.

ii. Reengineering and reverse engineering

There are some components of the software that had to be modified so as to reconstitute it in a new form an meet some of the specifications presented by the consumers. We made a provision for reengineering so that it could be easy for making minor changes in the system without carrying an overhaul of the whole system. We also used reverse engineering so as to make some further improvements into the system.

Mathematical and statistical models and techniques

We used the game theoretic model which is a form of a mathematical model for the design and development of the software. The models made it possible for us to develop a system that could meet the stated specific goals.

We also used some statistical model for obtaining the relationships between the variables used inform of mathematical equations (Bender, 2009).

Software engineering ethics

There is a number of software engineering ethics that were used in the implementation of the design. The ethics were observed so as to ensure that there were no conflicts that arose as a result of the design process between the developers and the customers and between the developers and law enforcement agencies.

We observed the following set of ethics when working on the product:

i. Agreeing to accept full responsibility for the work produced

ii. Bringing about a moderation between the interest of the developers and the customers

iii. Approve only the software that is considered safe

iv. Disclose the appropriate persons involved in the design process and expose any danger that may arise as a result of using the software

v. Provide services only in the areas of the developers' profession and be honest whenever delivering the services

vi. Strive for the production of the best quality products

Work Cited List

1. Brian Berenbach, Daniel Paulish, Juergen Katzmeier, Arnold Rudorfer Software & Systems Requirements Engineering: In Practice. New York: McGraw-Hill Professional. 2009

2. Adèr, H. J. Modelling. In H. J. Adèr & G. J. Mellenbergh (Eds.) (with

contributions by D. J. Hand), Advising on Research Methods: A consultant's companion (pp. 271-304). Huizen, The Netherlands: Johannes van Kessel Publishing. 2008

3. Bender, E. A. An Introduction to Mathematical Modeling, New York : Dover. 2000

4. Laplante, Phil Requirements Engineering for Software and Systems (1st ed.). Redmond, WA: CRC Press. 2009

5. Walter Sobkiw. Sustainable Development Possible with Creative System Engineering. New Jersey: CassBeth. 2008

6. Wiegers, Karl E. Software Requirements. Redmond, WA: Microsoft Press(2003).