ISSN: 2278 – 7798
International Journal of Science, Engineering and Technology Research (IJSETR)
Volume 5, Issue 4, April 2016

A Literature Survey on Software Clone Testing

Swati Sharma[#1], Priyanka Mehta[#2]
[1]M.Tech Scholar,
[2]Head of Department,
Department of Computer Science & Engineering,
Universal Institute of Engineering & Technology,
Lalru, Punjab, India

***Abstract*—Software Engineering deals with building the software in a systematic manner. Sub disciplines of software engineering include Software Requirement, Software Design, Construction, Testing, Maintenance etc. This paper gives the brief introduction about software Engineering, Software testing and various approaches of Software Testing, various types of Software testing. The main goal of this paper is to analyze software clone testing in brief including its pros and cons and discuss about the cloning process in detail. The various techniques of clone testing are also analyzed.**

***Index Terms*—Software Engineering, Software Testing, Software clone, clone detection, clone testing.**

I. INTRODUCTION

Software engineering deals with building, evolving and maintaining software systems in a systematic manner. Software engineer is responsible for handling software engineering projects that create, build software and tells its behavior [1].Software means computer programs. Software Engineering is the branch of computer science which means to create, operate, modify and maintain software components [10].

A. Software Engineering Sub-disciplines

Fig 1 shows the various disciplines of software engineering that are described as follow:

1. Software Requirement

A requirement specification is the complete description for the behavior of the system**.**

2. Software Design

It means process of problem solving and plan for solutions to the various problems. Dataflow Diagrams and Flowcharts, are comes under software design. In this we transform SRS document into design form using some tools.

3. Software Construction

In this we transform the design units into code segments. In the construction process we can define methods of the process and its description. It helps to improve software quality.

4. Software Testing

In this we test the various coding units. It checks whether the expected results match with the actual results. It is a process to identify correctness, completeness and effectiveness of computer software.

5. Software Maintenance

It is the modification of the software products for their correction after the delivery to correct faults, error and bugs in software or to adapt the system according to environment. It is a changing of software after delivering to the customer.

6. Software Configuration Management

It provides the auditing, changes and report to the changes that are made.

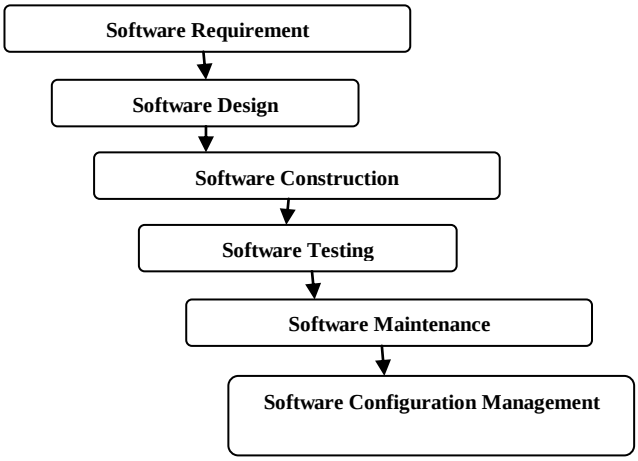

Fig 1: Disciplines of software engineering

II. SOFTWARE TESTING

Software testing means to find out error or faults in a system to make it correct, complete and to identify its quality. Software testing is used to detect the bugs in the system. Software testing is different from software development. It is part of software development [2]. It is an internal part of software development and closely related to software quality. The main aim of software testing is to make the system error

free. So software testing is mainly to find outs the error or bugs to improve the quality of the system [3]. Testing is a process of analyzing behavior of the product to detect difference between existing products and required conditions. It is a process to identify completeness, correctness and effectiveness of computer software. It checks whether the expected results match with the actual results [8]**.** It is the last phase of the product before deliver to the customer. Software testing is an important phase of software development which tells whether the product is efficient and error-free, work according to the requirements of the customer.

A. Software Testing Terminologies

The basic terminologies are as follows:

- Error: A mistake in coding is known as error. It is a misunderstanding of the internal stage.
- Defect: It is detected by the tester. It is some trouble in internal and external behavior of the products.
- Bug: the defect which is accepted by the developer and does not fulfill users requirement
- Fault: The invalid step taken which cause problem in future and give improper results [19].
- Failure: the result of fault is known as failure. The system is unable to perform according to the given specification.

B. Objective of Software Testing:

The main objective of software testing is as follows:

- To detect bugs or errors in a system.
- To improve the quality of a system.
- To prevent the system from the errors or bugs.

C. Complete Cycle Of Software Testing

Fig 2 illustrates the complete cycle for Software testing which includes various phases such as Requirement analysis, High level Design, Low level Design, Coding, Unit Testing, Integration Testing and System Testing.

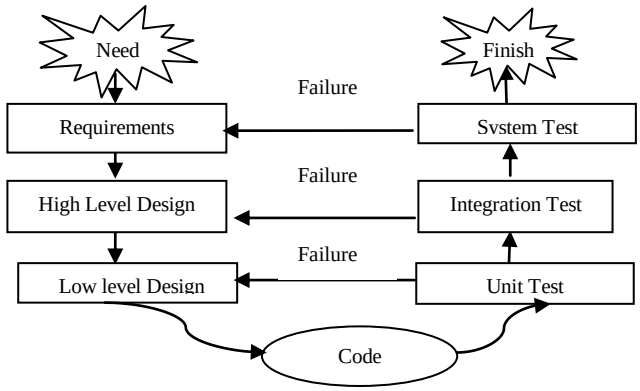

Fig 2: Complete cycle for testing

D. Approaches of Software Testing

There are three types of approaches which are followed by the software testing. These approaches are as following:

1. Bottom-Up Approach

Bottom–up is a one of the approach of software testing. It comes under integration testing. It tests the unit modules first such as programs and module and complete until the final phases reach. The control flow moves to upward direction. In bottom up approach of software testing, testing is conducted from sub module to main module [9].The main advantages of Bottom-up Approach are as following:

- Test results are easily observable.
- Easily test cases are generated.
- Faults are occurring at the bottom of the module only.

Disadvantages of bottom-up approach are as following:

- Driver modules are required.
- After the addition of last module only than program can exist.

2. Top-Down Approach

It is the second type of testing approach. In this approach we start from top level to bottom level. The control flow moves from top level to lower levels. In this, we firstly test the whole system first and then testing is applied to its sub modules. Then sub module are further tested and broken down into super sub modules [10].The main advantages of Top-down Approach are as following:

- Test cases are easily created after the I/O functions are added.
- If faults are find out at the early stage then it is useful.

The main disadvantages of Top-Down Approach as follows:

- Stub is complicated to produce at early stage.
- Observation of test output is difficult to produce.
- Testing and design are overlapped.

E. Types of Software Testing

There are different types of software testing are available to find out the defects in a software. These testing are as follow:

1. White-Box Testing

White-box testing is also known as Glass testing and structural testing. In white-box testing code is visible. The tester has knowledge of the internal mechanism of the components [4]. White-box testers are aware about the internal structure and also know how code is looks like. It is used in the validation process. It is a clear box testing because code can be easily visible in this type of testing.

2. Black-Box Testing

Black-box testing is also known as functional testing. This testing ignores the internal mechanism of the system. Testing which is based upon the output and having no knowledge of internal code [10]. It is testing in which its working is not understood by its user. It has no knowledge of processing of code but only concentrate upon the output. It is used in the validation process. It is based upon the requirements and functionality. There is no user requirement in this type of testing.

3. Unit Testing

It is a type of white-box testing. It is a testing for low-level design code. It is done within a class and starts from a individual module [4].It has testing the smallest unit of elements of a software which module or component or unit .It is basically done by programmer not tester and require detail of structure of code. It helps to design test drivers.

4. Integration Testing

This testing is done when two or more modules are combined together into a larger module. It verifies the functionality of the module after integration. It is done at the interfaces of the both the structure and component module. This type of testing is done in distributed or client/server modules. It uses both white box and black-box techniques [4].

5. System Testing

It is also known as end-to-end testing. It tests the complete application of environments. It is based upon the specification and requirement of the system. It checks the entire systems. It is for high level design and comes under the black-box testing [4].It also checks non-functional requirements also.

6. Load Testing

Load Testing is also comes under performance testing. It is done to determine whether the system is able to handle given no. of users or not. It is done only to check the performance of the system is it working good or not. It checks the behavior under heavy loads and inputs. In web application testing it is used to check where the performance of the system degrade and at which point it fails [5].

7. Sanity Testing

It is a testing which is done at the initial phase to check whether the new version of software is good enough to perform major testing later [5]. If the software is crashes at the earlier stage then further testing will not performed over it.

8. Acceptance Testing

Acceptance testing is a testing which is done by the user or customer not done by the developer. It is done to check whether it fulfill the user requirement or given specification. It checks the behavior of the system against user requirements [5].

9. Usability Testing

It is user friendly application testing. In this type of testing new user can easily understand its functionality. Proper user help is providing in each level of testing. It provides effective functionality for the end users.

10. Security Testing

It is also known as penetration testing. It is used to protect from the unauthorized users and hackers. It ensures that only authorized person can access the function of this system. It is used to find out weakness of the system which further may arise major harm for the system.

11. Reliability Testing [1]

Reliability testing is used to find out the point where the system fails and remove it before deployed. Its aim is to restart system after verification from major crashes. Estimation Model is used to analyze the present reliability and predict the future reliability with the help of it. Decision will take place with the help of estimation model in reliability testing whether the software will release or not by developer and whether it will adopt by users or not.

12. Regression Testing

Regression testing is a testing that refers to that section of the test cycle in which programs are tested to make sure that changes do not affect features that are not supposed to be affected. The process of verifying the modified software in the maintenance phase is known as Regression testing. Time and budget constraints are its major disadvantage due to complex process [11].

13. Clone Testing

Software engineering is an approach to the development, design operation and maintenance of software. Consequently in software engineering main focus is on to assure the quality in the product, detect the bugs and prevent system from bugs by testing or analysis. While developing any software for saving time and effort, software developer copy and paste program code again and again. So if any bug found in one module is reproduced in every copy. There are many copies of code present and no record of such copies is present [13]. This will make hard to fix such bugs and maintenance of existing software. Code clone is one of the factors making software maintenance more difficult [6].

III. SOFTWARE CLONING

Software clones are the fragments of source code which are highly similar; these similar fragments are called clones, clone classes, or clone pairs. Reasons of the Software Cloning may include intentional copying and duplication of code by programmers; clones may also arise due to automatically generated code, or due to the conditions imposed by the use of a particular framework or due to programmer's behavior such as laziness and the tendency to repeat common solutions, technology limitations, code understandability and external business forces have influences on code cloning. Cloning means unnecessary duplication of data whether it is at design level or at coding level.

By analysis of software application it appears that these clones results from the addition of some extra functionality which is similar but not identical to some existing logic within a system Code Clones are the code segments that have equal functionality. A code clone is nothing a similar or duplicate code in a source code or created either by replication or some modifications. [7] These are the parts of code which will give same output when same input is given. It is important to detect these code clones to determine bugs and for reusing the old software as they may harm the software maintenance. These Code Clones may lead to higher maintenance cost because modification will be required several times. These cloned codes add to high

maintenance cost of software and also cause the code bloating. This is because when changes perform on one clone, then the same action is performed on respected clone, this will increase the maintenance. These clones can also increase risk of in faults in system by increasing the risk of making inconsistent changes to the code [7]. All similar pieces of code should have to be checked for same bug, if there is presence of bug in any piece of code. Additionally, when we fix these cloned code bugs, it will lead to further system errors. Clone detection is an important task of software analysis, so it is very important to extract these clones to perform a number of software engineering tasks such as plagiarism detection, software evolution analysis, program understanding (clones may carry domain knowledge), code quality analysis (fewer clones may mean better quality code), copyright infringement investigation aspect mining (clones may indicate the presence of an aspect), copyright infringement investigation, code compaction (for example, in mobile devices), virus detection, and bug detection. Previous studies have shown that around 7% to 23% of the source code in a software system contains code clone [7].There are number of tools to detect the code clones, but it is not effective to remove the clones. Because code clones are needed for software to function properly. So we can apply the principal of refactoring or modularity to improve the reusability and maintainability of software from clone code.

A. Reasons for Software Cloning

The practice of code duplication is considered as bad for activity for maintenance phase however the programmers prefers this because of many reasons. Fig 3 shows the reasons for software cloning and are mentioned below:

Fig 3: Reasons for Software Cloning

1. Time limitations

One of the major causes of code cloning is that a certain time limit is assigned to developer to finish a project. To do this, developer just copy and paste the existing code and adapt to their current needs.

2. Limited Skills of Programmer

Due to limited skills set of the programmer, he will copy/paste/edit the existing code.

3. Performance of Developer

Sometimes the performance of programmer is measured by the number of lines he produces per hour. In such cases, the programmer's focus is to increase the number of lines of the system and hence he tries to reuse the same code again and again by copying and pasting.

4. Difficult to understand complicated systems

The large systems are very complex and difficult to understand. The programmers are bound to copy existing functionality and logic in the programs.

5. Language Constraints

Clones can be introduced due to limitations of language, especially when the language does not have sufficient abstraction mechanisms. Sometimes the developers are forced to copy because of limitations of their knowledge in that particular programming language.

6. Uncertainty of fresh code

It is an easy to copy the code rather than implementing new ideas in existing software. The Programmers have the fear that new code may result in difficult and lengthy software development life cycle. Furthermore new code may lead to introduction of new bugs.

7. By Coincident

Code cloning may be accidently. There may be case that two software developers may come with same solution. Programmers may unintentionally repeat a common solution for similar problems. Therefore, several clones may unknowingly be creates to the software systems.

B. Advantages of Clones

The introduction of clones in software is deliberately done by software developers a study was conducted on this issue by Kapser and Godfrey [11, 12]. Some of the advantages of introduction of clones in software are mentioned below:

- Changing requirements can be accommodated easily and in a fast manner.
- The use of templates is encouraged in some programming languages thus making the task easier for the programmer.
- Lacks of reusability and abstraction in a programming language can be easily fulfilled with the help of code clone
- For making the system efficient the overhead of procedure calls can be compensated using code duplication.

C. Disadvantages of Clones

In this subsection we will be introducing few disadvantages of clones 4.

1. Higher maintenance costs

The activity of copy and paste increases the maintenance cost of the software, two studies [13, 14] confirm the same fact.

2. Bug propagation

If a bug is present in a code segment and the same code segmented is pasted at different parts of the program, it is quite evident that the same bug will be present in all the parts of the software. Hence increasing the possibility of bug propagation [15, 16].

3. Bad impact on design

Code duplication does not support the activities like refactoring, inheritance, abstraction etc. [17, 18]. Hence leading to a bad design.

4. Impact on System Understanding/improvement

It is quite common that the person who developed the original system is not the one who is maintaining it.

D. Advantages of Clone Detection

1. Detect Library Candidates

It has noticed that a code fragment that has been copied and reused multiple times in the system apparently proves its usability. As a result, this fragment can be incorporated in a library, to announce its reuse potential officially.

2. Bug Detection

Copy-pasted software bugs, especially, can be successfully detected by clone detection tools.

3. Program Understanding

Clone Detection Techniques may assist in understanding a software system. As clones hold important domain knowledge, one may achieve an overall understanding of the entire system by understanding clones of a system.

4. Find Usage Patterns

If all the clone fragments of a same source fragment can be detected, the functional usage patterns of that fragment can be discovered. This is the reason why the software clone detection gains considerable attention.

E. Clone detection techniques

Clone detection techniques are roughly classified into four main categories: textual lexical, syntactic and semantic.

1. Textual approach

In Textual approaches there is little need of normalization or transformation of code. In this basically line to line comparison is done, which basically based on two type one is simple line matching and other one is parameterized line matching. This technique is basically string based.

2. Lexical approach

In lexical technique we convert source code into tokens using lexical rules. These tokens are then compared.

3. Syntactic approach

In syntactic technique an abstract tree is generated. Using parser source code is converted into parse tree. Abstract tree is then processed either using tree matching or metric to find the clones.

4. Semantic approach

In semantic approach a source code is represented a program dependency graph. Nodes represent the statements and expressions and edges represent control and data dependencies.

A clone detector tool must try to find pieces of code of high similarity in a system's source code or text. The main problem is that, it is not known initially which code fragments may be repeated. Thus the detector really should compare every possible code of fragment with every other possible fragment. Such a comparison is expensive from a computational point of view and thus, several measures are used to reduce the domain of comparison before performing the actual comparisons. Even after identifying potentially cloned fragments, further analysis and tool support may be required to identify the actual clones. In this section, an overall summary of the basic steps in a clone detection process is provided. This generic overall picture allows us to compare and evaluate clone detection tools with respect to their underlying mechanisms for the individual steps.

F. Clone Detection Process

A clone detector must be able to find parts of code of high similarity in a source code of a system and it is a known fact that we cannot judge which code fragment can be repeated hence an efficient detector should compare each code fragment with others. This one to one comparison is very expensive in terms of computation. Therefore many measures are taken to reduce the domain of comparison. After the identification of the clones there can a need for additional technique or tools for retrieving the actual code. Figure shows the set of steps that a general clone detector may follow. The steps listed can be a part of clone detection technique however it is not necessary [12]. We are providing the basic steps in a software clone detection process and all the steps are not included in all the techniques. The details of the phases of clone detection process are given below:

1. Pre-processing

The source code of divided into code fragments and the comparison domain is determined. The main objectives of this phase are :

- All the unnecessary source code is deleted.
- The remaining source code is divided into set of code fragments.
- The fragments are further processed in accordance with the comparison technique used.

2. Transformation

Once the decision of units of comparison is made, the source code has to be transformed to an intermediate representation or in other words extraction of units from the source code need to be done. This process can also be related to extraction in reverse engineering field.

3. Extraction

We have to transform the source code to a suitable form which can be accepted as in put by the comparison algorithm.

This method involves production of tokens, parse tree and control and data flow on source code.

4. Normalization

Normalization is used to remove white space, difference in commenting and formatting and normalizing identifier names. Pretty Printing is also done in this phase in addition to all the structural transformation.

5. Formatting

The output of the comparison algorithm which is the clone pair list is transformed to a clone pair list that is acceptable by the original source code database. The mapping of coordinates of clone pairs to the original source file is also done.

6. Post processing/filtering

The ranking of clones or filtering of clones are done in this phase. The method used for filtering can be manual analysis or heuristic approach.

IV. CONCLUSION

In this paper, Software Cloning is being reviewed. Also advantages, disadvantages and reasons of software clone testing have been discussed. Clone Detection process have also been discussed. Some of the techniques of Software Clone testing are discussed.

REFERENCES

[1] http://www.slideshare.net/engineerrd/software-requirement
[2] Salwa K. Abd-El-Hafiz , "*Code Cloning: The Analysis, Detection and Removal*" , International Journal of Computer Applications (0975 – 8887) Volume 20– No.7, April 2013.
[3] Swarnendu Biswas and Rajib Mall, "*Regression Test Selection Techniques: A Survey*", Informatica 35 , 289–321, April 2011.
[4] Dr. Varun Kumar, Sujata, Mohit Kumar, "*Test Case Prioritization Using Fault Severity*", IJCST Vol. 1, Issue 1, September 2010.
[5] Sanjeev Chakraborty, "*Code Clone detection: A approach*" Polaris Software, 2010.
[6] Myers, Glenford. (1979). The Art of Software Testing.
[7] Ashall, F. N. "*Testing The Product Propagation*" ,2011.
[8] http://guru99.199tech.com/software-testing-introduction-importance.html
[9] Angad Singh Gakhar , "*Converting Code Clones To Aspects Using Algorithmic Approac*" , 2009.
[10] W. Eric Wong, J. R. Horgan, Saul London, Hira Agrawal, "*A Study of Effective Regression Testing in Practice*", IEEE, 2008.
[11] S. Bellon, R. Koschke, G. Antoniol, J. Krinke and E. Merlo, "*Comparison and Evaluation of Clone Detection Tools*", Transactions on Software Engineering, 33(9):577-591 (2007).
[12] Stefan Bellon , "*Detection of Software Clones Tool Comparison Experiment* " , presented at the 1st IEEE International Workshop on Source Code Analysis and Manipulation, Montreal, Canada, October 2002.
[13] Sonam Gupta, P. C. Gupta, "*Literature Survey of Clone Detection Techniques*" , International Journal of Computer Applications, Volume 99 - Number 3, Year of Publication: 2014.