Test Case Quality Management Procedure for Enhancing the Efficiency of IID Continuous Testing

Sen-Tarng Lai*
Department of Information Technology and Management, Shih Chien University No.70, Dazhi St., Zhongshan Dist., Taipei City 104, Taiwan (R.O.C.)

*Corresponding author. Tel.: +886-2-25381111 ext. 8954; email: stlai@mail.usc.edu.tw
Manuscript submitted June 29, 2017; accepted August 4, 2017.
doi: 10.17706/jsw.12.10.794-806

Abstract: Software development process should have continuous adjustment and modification ability to meet the multifaceted needs of the users. Iterative and Incremental Development (IID) in accordance with user requirements to complete an incremental workable version, then test and assessment. However, new versions iterative testing and release need to spend much time and resources. It is the critical challenge that IID has to overcome. Continuous integration environment and high quality test case can increase the efficiency of IID continuous testing (CT). For this, in this paper, CI main activities and test case quality factors are deeply investigated. In order to improve the efficiency and quality of CT activities, test cases should have qualified documentation, manageability, maintainability and reusability quality characteristics. According to the CT activities, the paper proposes the test cases quality measurement (TCQM) model for test case quality improvement. And based on the TCQM model, designs the test case quality management procedure to timely identify and control test case quality defects and effectively enhance the activities efficiency of IID CT.

Key words: Continuous testing, IID, quality measurement, test case.

1. Introduction

According to the Standish Group software project survey report in 2009, only 32% belong to the successful software projects that development time, budget and requirement quality almost fully met the project planning. In development process, 24% projects were canceled and the rest 44% owing to development time delay, over budget and quality unsatisfied the requirement situations were regarded as the failure projects [1]. In 2014, the latest research report of the Standish Group further pointed out that 31.1% of software projects are canceled, the development cost of 52.7% canceled projects exceed 189% of the original budget. Investigating the major cause of software project failure is high frequency of plan change. In software development process, many unexpected events will occur successively. The events not only impact the software development activities but also causing to the software project plan must change continuously. However, project plan modification often is not able to effectively and timely reflect to the important plan items that include budget planning, development time, quality management and resource allocation. Therefore, plan changes make the gaps between plan items and the software development process. The gaps gradually cause to the project plan completely unable to control the budget execution, development time, quality management and resource allocation. Finally, software project becomes a failure project of budget overruns, schedule delays and poor quality [2]-[4]. T Therefore, in order to handle the

changeable requirements of software project, software development process must have the extensibility and changeability to reduce the risk of function extension and requirement change.

Internet widespread age, the pursuit of a variety of high-efficiency and high-performance trading activity must be combined with the appropriate network. In order to overcome the ever-changing hardware facility, advanced IT technology, services requirement change and continuous expansion functions, software projects should carefully select the appropriate development model. Agile development model uses IID (Iterative and Incremental Development) process, each repeated development work required to complete a new system version and immediately test and evaluate this version. Early detection the processes problems and product defects can timely proceed the improvement measures and reduce the risk of software development [4]-[6]. IID model gradually being taken seriously, more of the software development models have recommended to combine with IID process. Among which the CI (Continuous Integration) with agile development model most cited and discussed in [7]-[9]. CI is a software implementation concept that was proposed in 1998. CI have combined testing tools, analytical tools, software version control system and other aspects of technology and become an important member of the agile development model. The advantages of CI can identify software development problems and product quality defects in software initial development stage to reduce the risk of software development [4], [7]-[10].

Information technology and application rapid evolution makes software project must have the flexible and changeable features. Therefore, software development process is very suitable to use IID methodology and apply CI practical concept. In early development phase, coding errors and quality defects can be quickly identified, corrected and revised that make the risk of software project can be greatly reduced. CT (Continuous Testing) of IID can timely identify the process problems and quality defects. However, it need consume more time and human resources which is a major defect of IID. Applied test case quality management, CT can effectively compensate for the drawback of IID process. In this paper, CT of IID and test case advantages are discussed. Analyzing key items and quality characteristics of test case, high quality test case to make up the IID process defects. This paper proposes the Test Case Quality Measurement (TCQM) model, and based on the TCQM model develop the test case quality management procedure to assist CT activities of IID. Unit programs can be tested automatically, integration and acceptance testing can be accomplished quickly and new version software can be deployed on time. In Section 2, software project failure factors and reducing software development risks are discussed. In Section 3, IID CT activities and test case importance are investigated. In Section IV, discusses the major quality factors of test case and proposes the test case quality measurement model. In Section V, based the TCQM model designs a test case quality management procedure. In Section VI, emphasizes the importance Test Case Quality Management Procedure for enhancing CT activities efficiency, and does the conclusion for the paper.

2. Software Project Risks and Agile Software Development

Software development risk is major factor to cause the software project failure. IID process enhances refactor ability and changeability to effectively overcome the critical software development risk.

2.1 Major Failure Factors of Software Project

According to the Standish Group survey report of large scale software project, software project success ratio is under 30%. And 80% of failure projects are time delay, over budget and quality not satisfied the requirements [1]. Many papers have discussed the causes of software project failure [11]-[14]. Development time delay, over budget and quality unsatisfied the requirements etc. three situations always cause to software project with high development risk. The reasons of the high risk is high relationship with the following four key events:

1) In the early stages of software development, system and requirement analyst collect and analysis

incomplete information to cause system requirement specifications and user requirements existed incorrect, incomplete and inconsistent defects. The defects can't be revised immediately. It is sure to cause the bad situations which include development time delay, over budget and product quality unsatisfied the requirements.

2) In software development process, new technology or development environment evolution makes the project plan must to be adjusted to match new technology and development environment. Before planning items unable be revised immediately, the follow development operations will be affected.
3) In initial development stage, user submits many kinds of immature requirements. The submitted requirements often are requested to expand, modify, or remove in software development process. Frequent requirements changes become the major impaction for the follow phase development.
4) Each software development phase need many resource items that include developers, hardware devices, software tools, and development environment. Incomplete resource allocation may cause the project plan resource management need to readjustment and allocation. Imperfect and inflexibility resource management often makes high risk of software project development.

Four major risk events of software project can summarize as incomplete requirement analysis, new technology and environment evolution, frequent requirements change, and Imperfect and inflexibility resources allocation management. In software development process, these high risk events almost cannot be avoided or excluded. Before risk evented occurred, planning the risk events prevention and detection approach becomes the best way to reduce the software development risks.

2.2. Agile Software Development and IID

In recent twenty years, the information technology, development environment and user requirements, and software development methodologies continuously evolve. Most of software development models have high relationship with user requirements. Waterfall model defines clear phase mission and very concerns the phase documents. The quality of requirement documents can't reach correctness, completeness and consistency, development process will be denied to enter the following phase. Recently proposed development models have modified requirement specification style. The requirement items are allowed to propose by incremental manner, not necessary to decision at same time [15]-[18] Using iterative development model, user can incrementally provide the requirement items. Software development change risks can be greatly reduced. Each development model should has the adjustment strategy to handle the change of user requirements. The adjustment strategy can't effectively apply to reduce software development change risks, then may impact success ratio of software project.

In 2001, seventeen software developers met at the Utah of USA and produced the Manifesto of the agile software development. Many of participants had previously authored their own software development methodologies, including Extreme programming, Crystal, and Scrum [4], [6], [15]. Agile software development proposes several critical viewpoints:

- In development process, does not concern analysis and design phase operations and documents.
- As soon as possible to enter programming phase, workable software is more practical than development document.
- Support and provide high changeability software system.
- Enhance the cooperative relationship between developer and user, user can fully attend the development team.

Agile software development neglects the formal analysis and design phases, uses non-document oriented development, and doesn't care the follow-up maintenance operations. These are the major challenges of

agile development. For effectively reducing the software development risks, the challenges of agile development should concretely overcome. However, agile software development uses IID to reduce user requirements complexity, refactoring feature increase the requirement modified flexibility, and non-document oriented can reduce the requirements change cost. Agile software project must release a workable version during two or three weeks. Letting the client can detailed test, validate requirement items of new version and clearly understand the development progress. In each day, fifteen minutes stand up meeting is held to effectively reach full intercommunication between client and developers. In development change impacts, IID greatly decreases schedule delay, cost over budget and quality unsatisfied user requirement situations, software development change risk can be effectively reduced. However, continuous version integration test need expend more manpower and resource that is major drawback of IID. CT activities of IID need high efficiency and high quality operations mechanism to reduce another development risk.

3. Continuous Testing and Importance of Test Case

Combining high quality test cases with CT activities can increase CI quality and productivity.

3.1. CT Activities of IID

The major purpose of the CT activities is to ensure that the quality of the implementation phase meets the phase specifications. Agile software development methods use the IID process for incremental requirements. Therefore, the CT activities become the critical task of agile development methods to challenge the time management. For reducing the manpower and resources investment of CT activities, agile software development simplifies the test operations into three to four steps, and uses the automated testing tools to improve the quality and efficiency of CT activities. In agile software development, CT activities use unit testing, integration testing, system testing and acceptance testing four steps to ensure the delivery software quality (shown as Fig. 1). Four testing steps are described as follows:

- Unit testing: Unit testing is the first and primary test step in the test operations. All module units or class components must pass unit tests before being released for integration. Complete test data, expected results and suitable unit testing tools can reach automated unit testing.
- Integration testing: After unit testing, the related module units combines into the function group to integration testing. Extensive integration testing also considers the integration of internal and external entities. Combining the existing unit test cases, part of integration test cases can be generated.
- System testing: A process of attempting to find discrepancies between the system and its original objectives. According to the system requirement specification, system test cases need to consider functional and nonfunctional test cases. Collecting the related system test script can assist the automatic system regression testing.
- Acceptance testing: New version of software must pass acceptance test with clients before release for delivery. Based on the user requirements, client representative attend to verify the requirement items one by one in required working environment. Acceptance test cases have high flexibility and depend on the client representative operation behavior.

For overcoming the CT activities predicament of IID, three critical tasks should achieve concretely:

1) Unit testing must fully automate.
2) Each level regression testing should have automatic.
3) Selecting a high efficiency method for integration testing.

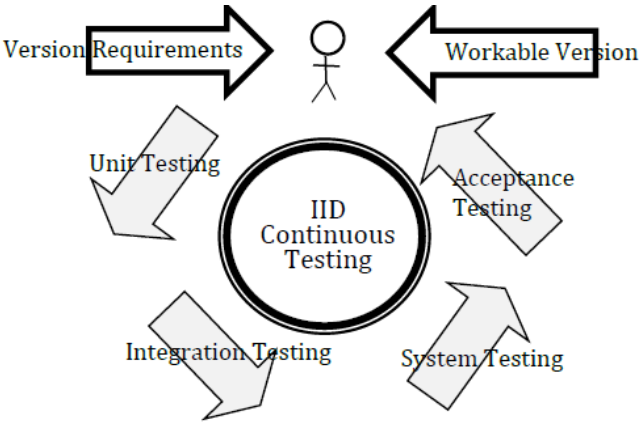

Fig. 1. Major activities of CT.

3.2 Importance of Test Case

In order to concretely reduce software development risk, CI should play a critical role to continuously monitor coding errors and process defects. CT activities must suitably combine automated tools and quality control procedures to reduce resources expense and person involvement. However, CT activities efficiency must depend on the test case quality. Test case is an important assets in software development especially in software test activities. The major purpose of test cases is to find the undiscovered code errors or defects. The other, test case should have the attributes to improve CT activities quality and efficiency in IID process. Therefore, test case need accomplish the four major tasks:

1) Identifying the undiscovered code errors and defects to improve software quality.
2) Unit test cases can be extended to integration and acceptance test cases.
3) Accomplishing automated unit testing and regression testing to increase CT activities efficiency.
4) Along with the code, design and requirement changes, test cases should have changeability and preserve the difference between the versions completely.

For reaching four tasks, test case must implant critical quality characteristics which include the qualified documentation, manageability, maintainability and reusability.

- Qualified documentation: Test case document is the necessary items for improving test activities quality and effects. Uniform test case items specification can assist generate practical test cases for automated unit testing. Correct, complete and consistent test case document can assist to increase CT activities quality and efficiency.
- Manageability: IID process emphasizes modification flexibility and scalability. In development process, code, design and requirement items can be flexibly changed or adjusted. Like other development documents, test case must be flexibly changed or adjusted and easy retrieval and recovery. Therefore, test case version control becomes a critical mechanism to handle CT of IID process.
- Maintainability: IID need invest more manpower and resources for handling CT activities. CI environment uses the automated testing tools and integration function to mitigate the problems of manpower and resources. However, simple, changeable and traceable test case is the more important factor for improving CT activities quality and efficiency.
- Reusability: Reusability: In IID process, code, design and requirement items can be flexibly changed or adjusted. Requirement changes or adjustments make the regression testing and retesting need to be handled frequently. In addition, CT activities need to include unit testing, integration testing, system testing and acceptance testing four levels. Some basic level test cases can assist to combine

into the higher level test cases. Therefore, test case should have the reusability for improving CT activities quality and efficiency.

4. Test Case Quality Measurement

Test case quality is important critical item to affect CT activities efficiency. Therefore, in this section discusses the major quality factors of test case and proposes the test case quality measurement model.

4.1 Critical Quality Factors of Test Case

For improving the quality of test case, four quality characteristics should be concerned:

- Documentation quality: Test case documentation is the necessary items of test case generation, change, revision, integration and reuse. For reaching the test objective, test case document should have uniform standard format, correct, complete and consistent quality features.
- Management quality: In IID process, user story can continuously and incrementally submit for extending the new version. Therefore, in order to enhance the efficiency of CT activities, test cases should have the manageable features for easy retrieval, quick recovery and version control mechanism. Therefore, management quality should have three characteristics:
  - Easy retrieval the different levels test cases (Test case easy identification),
  - Quick recovery the previous versions (Recoverability),
  - Detailed record the revision date, reasons, tester and related activity (Version control quality).
- Maintenance quality: Test case is an important document for improving software development quality and efficiency. In IID process, test case must keep high maintainability to increase CT activities efficiency. Maintenance quality of test case should have three characteristics:
  - Simplicity: simple test case can increase the maintainability.
  - Changeability: changeable test cases is a necessary condition for increase maintainability.
  - Traceability: cross-reference tables can enhance test cases traceability and maintainability.
- Reuse quality: CT activities include unit, integration, system and acceptance four testing levels. Some test cases of integration testing can refer the unit test cases. Some test cases of system and acceptance testing can refer the test cases of unit and integration testing. Reuse quality of test case should have three characteristics:
  - Automaticity: reusable test case can increase the percentage of automatic testing (e.g., unit testing, regression testing).
  - Extendibility: the base level test cases can expand to the higher level testing cases.
  - Combinability: Some higher level test cases need to combine the base level test cases to increase efficiency of CT activities.

4.2 Test Case Quality Measurement Model

For collecting useful quality factors, it is necessary to develop a set of complete and clear inspection checklists and conduct the test case quality audit. Checklists design and implementation can assist to collect the quality factors of test case. However, individual factor or measurement can only measure or evaluate the specific quality characteristic. In order to effectively monitor and assess test case quality defects, individual measurements should make the appropriate combination [19], [20], [21]. Two kind of metric combination models are Linear Combination Model (LCM) [19], [20], and Non-Linear Combination Model (NLCM) [20], [21]. NLCM has higher accuracy measurement than LCM. However, LCM has high flexibility, more extensible and easy formulation than NLCM. For this, in this paper, LCM is applied to test case quality measurement. The different level development activities have different quality metrics be shown. Therefore, before applying the linear combination model, the quality factors must be normalized. The normalized value is

between 0 and 1. The best quality quantified value approaches to 1 and the worst quality approaches to 0.

Test case should consider four quantified qualities that include documentation, manageability, maintainability and reusability quality. Using the LCM, related quality factors can be combined into the primitive metric, and then the related primitive metrics can be combined into the quality measurements. Finally, combines with four critical quality measurements and generates a test case quality indicator. Five formulas described as follows:

1) Document Quality Measurement is combined uniform format, correctness, completeness and consistency four metrics. The combination formula is shown as Equation (1):

DQM: Document Quality Measurement

UFM: Uniform Format Metrics	W_1*: Weight of UFM*
CrM: Correctness Metrics	W_2*: Weight of CrM*
CmM: Completeness Metrics	W_3*: Weight of CmM*
CnM: Consistency Metrics	W_4*: Weight of CnM*

$$DQM = W_1 * UFM + W_2 * CrM + W_3 * CmM + W_4 * CnM \qquad W_1 + W_2 + W_3 + W_4 = 1 \tag{1}$$

2) Manageability Quality Measurement (MgQM) is combined retrieval, recoverability, version control mechanism three quality metrics. The combination formula is shown as Equation (2):

MgQM: Manageability Quality Measurement

RQM: Retrieval Quality Metrics	W_1*: Weight of RQM*
RM: Recoverability Metrics	W_2*: Weight of RM*
VCM: Version Control mechanism Metrics	W_3*: Weight of VCM*

$$MgQM = W_1 * RQM + W_2 * RM + W_3 * VCM \qquad W_1 + W_2 + W_3 = 1 \tag{2}$$

3) Maintainability Quality Measurement (MtQM) is combined changeability, traceability and extendibility three quality metrics. The combination formula is shown as Equation (3):

MtQM: Maintainability Quality Measurement

SM: Simplify Metrics	W_1*: Weight of SM*
CM: Changeability Metrics	W_2*: Weight of CM*
TM: Traceability Metrics	W_3*: Weight of TM*

$$MtQM = W_1 * SM + W_2 * CM + W_3 * TM \qquad W_1 + W_2 + W_3 = 1 \tag{3}$$

4) Reusability Quality Measurement (RQM) is combined retesting, integrate ability and combinability three metrics. The combination formula is shown as Equation (4):

RQM: Reusability Quality Measurement

AM: Automaticity Metrics	W_1*: Weight of AM*
EM: Extendibility Metrics	W_2*: Weight of EM*
CM: Combinability Metrics	W_3*: Weight of CM*

$$RQM = W_1 * AM + W_2 * EM + W_3 * CM \qquad W_1 + W_2 + W_3 = 1 \tag{4}$$

5) Finally, combining DQM, MgQM, MtQM and RQM into a Test Case Quality Indictor (TCQI). The combination formula is shown as Equation (5):

TCQI: Test Case Quality Indictor

DQM: Document Quality Measurement	W_d*: Weight of DQM*
MgQM: Manageability Quality Measurement	W_{mg}*: Weight of MgQM*
MtQM: Maintainability Quality Measurement	W_{mt}*: Weight of MtQM*
RQM: Reusability Quality Measurement	W_r*: Weight of RQM*

$$TCQI = W_d * DQM + W_{mg} * MgQM + W_{mt} * MtQM + W_r * RQM \qquad W_d + W_{mg} + W_{mt} + W_r = 1 \tag{5}$$

In IID process of agile development model, test case quality is major cause to effect CT activities. For measuring and improving the quality of test case, the major quality factors of test case were collected and divided into 13 groups. In first layer, 13 groups basic quality factor are combined into 13 quality metrics. In second layer, 13 quality metrics are combined into four quality measurements. In third layer, four quality measurements are combined into a Test Case Quality Indictor (TCQI). With three layer combination formulas to generate a TCQI, the process is called a Test Case Quality Measurement (TCQM) model. TCQM model architecture is shown as Fig. 2. TCQI is a relative judgment mechanism also is a basis to identify quality defects of test case. In TCQM model, basic quality factors are combined into high layer quality measurements. According to the combination formulas, quality measurements construct by the 4 measurement equations and 13 quality factors. For improving test case quality, the related activities of affected quality factors should be rigorously inspected. Identifies the quality problems or defects, then proceeds improvement action.

5. Test Case Quality Management Procedure and Efficiency Evaluation

5.1 Test Case Quality Management Procedure

In each incremental development of IID, test cases are planned and designed according to new version requirements. For improving quality and efficiency of IID, the test case quality should be effectively managed and controlled. Test case quality management procedure will be triggered in each incremental development of IID. Quality factors collection and TCQM model can timely identify the test case quality problems and defects. Continuously improvement test case quality can concretely enhance CT activities efficiency and quality. The detailed inspection tasks of test case quality management procedure (shown as Fig. 3) describe as follows:

Step 1. Test case document quality inspection: Based on TDD model, after user proposes new version requirements, the unit test activities should be designed first [17]. Test case documentation with uniform specified items are inspected. Then, according to the new version requirements, the correct, complete and consistent contents of test case are inspected.

Step 2. Test case manageability quality inspection: In order to assure test case manageability quality, three items should be inspected:

- Each level of test case can be retrieved quickly by clear test case identification.
- The difference among the versions of test case must be able to clearly and quickly identify.
- Changed or adjusted test case version has the complete and detailed records in version control system.

Step 3. Test case maintainability quality inspection: In order to assure test case maintainability quality, three items should be inspected:

- For the simplicity characteristic, test case planning and design should consider the single purpose and easy to judge the test results.
- For the changeability characteristic, test item should have the modularity feature that has low coupling and high cohesion for reducing the test item complexity.
- For the traceability characteristic, planning and design test case also need to build the complete cross-reference tables with the related documents.

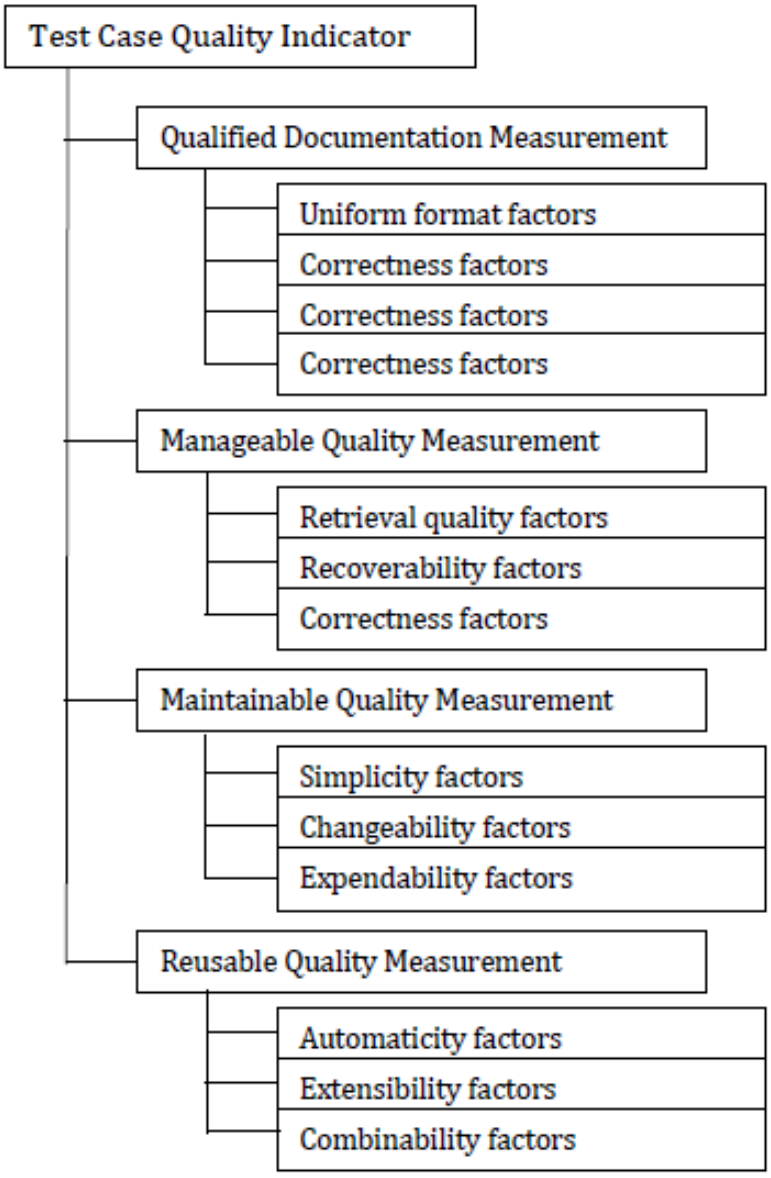

Fig. 2. The architecture of TCQM model.

Step 4. Test case reusability quality inspection: In order to assure test case reusability quality, three items should be inspected:

- For automatic characteristic, test case document that includes test data and test results must record as the format of testing tools.
- For extensibility characteristic, each level test case should have the ability of easy adjustment and modification to meet the extension requirements.
- Does the planning and design of test cases have combination features for incremental development?

Step 5. TCQM model quality factors and equations adjustment: In order to assure test case quality can improve the efficiency of CT activities, three items should be adjusted:

- According to the software developer suggestions and data collection, test case quality defects should be timely identified and suitably adjusted.
- According to the user software requirements and different software features, the weighted value of equations of measurement model should be timely modified.

- TCQM model continuous improvement can increase the efficiency and quality of CT activities.

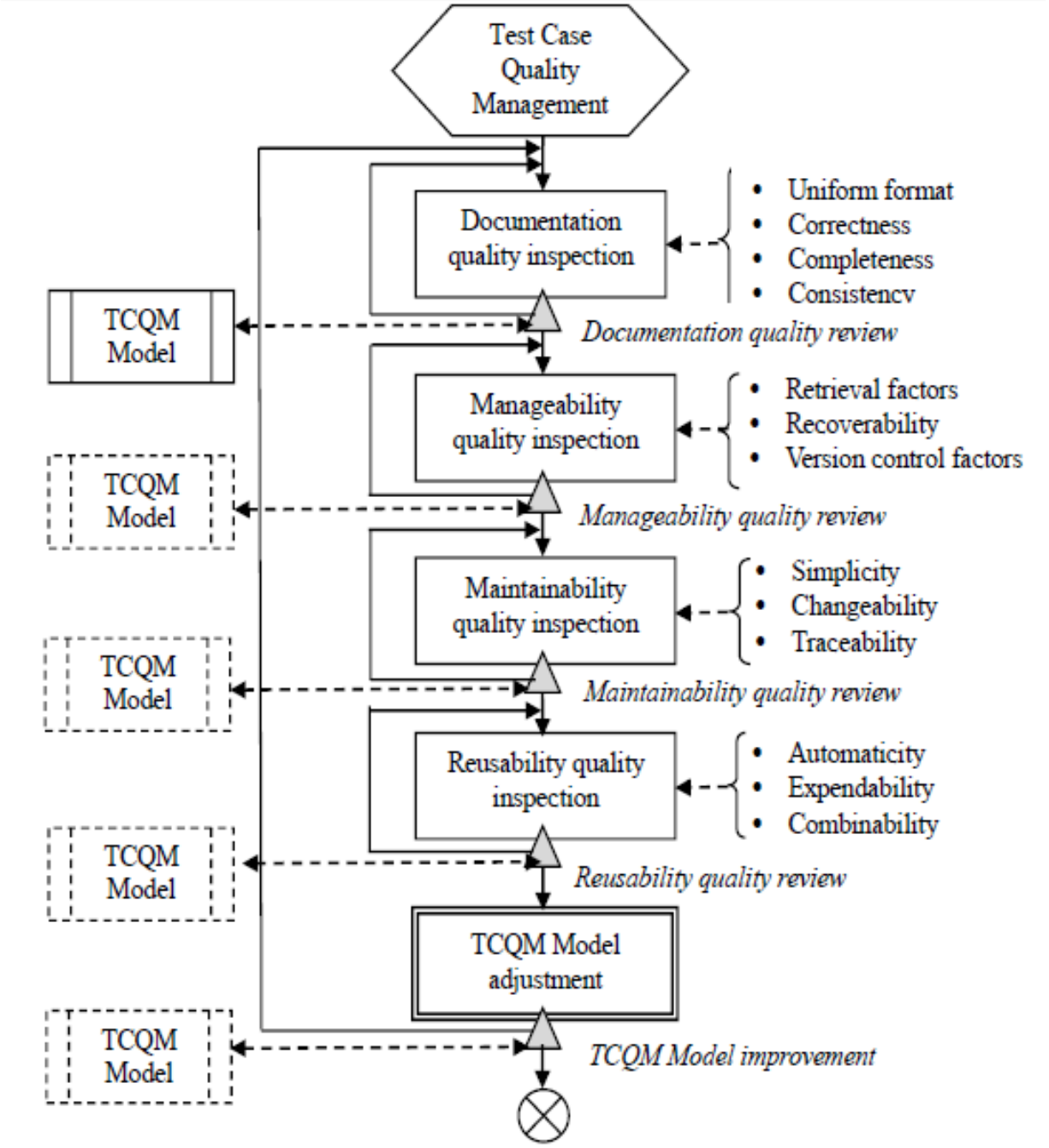

Fig. 3. Operation flow of test case quality management procedure.

5.2 TCQM Efficiency Evaluation

Test case is an important asset of software system. Test case must be forever and completely reserved until software system retirement [4]. For evaluation the advantages of TCQM model, three test case management approaches that include traditional control, version control and TCQM model are compared. In CT activities, test case must quickly complete retrieval, modification, extension, testing and retesting five tasks. Therefore, the comparison items are test case retrieval, modification and extension and test automation four tasks efficiency (shown as Table 1). The key to tasks efficiency comparison described as follows:

1) Retrieval task: TCQM model combines with the identification and cross-reference quality factors.

The related test cases or specific test case can be quickly and easily retrieved. The other approaches can easily retrieve the test case with clear identification.

2) Modification task: TCQM model combines with the document quality, manageability, and simplify quality factors. For any modification task, the test case can be effetely modified with high efficiency. The other approaches without consider test case quality, they need to take more resource for test case modification.
3) Extension task: TCQM model combines with the document quality, maintainability and version control mechanism quality factors. For the extension task, the test case can be timely extended in limited resource. The other approaches without consider test case quality, they need to take more resource for test case extension.
4) Test automation: In order to accomplish the test automation, test data and test results must save with the format of testing tools. The document quality of TCQM model can match to the format of testing tools. For unit testing and regression testing, the test case can completely coordinate with the testing tools and accomplish the test automation.

Table 1. Task Efficiency Comparison of Three Test Case Management Approaches

Management Approaches / Comparison Tasks	Traditional Control	Version Control	TCQM Model
Retrieval	*Middle*	*Middle*	*High*
Modification	*Low*	*Middle*	*High*
Extension	*Low*	*Middle*	*High*
Test Automation	*Low*	*Low*	*High*

6. Conclusion

In the rapidly changing network age, everything requests the achievement of fast. Software project must overcome the changing requirements and to withstand environmental challenges that continued to introduce new technology. Therefore, development process should have the ability to modify and adjust to meet the diversified needs of the users. However, software project successful rate always under 40%, how to effectively reduce the risk of software development project, is a topic worthy of further exploration. In order to effectively reduce the requirement change to cause software development risks, IID has recognized as the very important and practical development methods in current stage. Many software development methods and models have been taken IID process to reduce the risk of requirement changes. CT of IID can timely identify the process problems and quality defects. However, it need consume more time and human resources which is the main challenge of IID. Applied test case quality management, CT can effectively overcome the challenge of IID. In this paper, CT activities and test case quality factors are discussed. Analyzing the quality characteristics of test case, and applying high quality test case improve the efficiency of CT activities. According to the CT activities, the paper proposes the test cases quality measurement (TCQM) model for test case quality improvement. And based on the TCQM model, designs the test case quality management procedure to timely identify and control test case quality defects and effectively enhance the activities efficiency of IID CT. Unit programs can be tested automatically, integration and acceptance testing can be accomplished quickly and new version software can be deployed on time. Test case quality management procedure help developers achieve four results:

- Test case can actively accomplish automated unit testing.

- In CT activities of IID, the unit, integration and acceptance testing efficiency and quality can be improved.
- High quality test case can help accomplish automated regression testing of CT.
- Effectively reduce software project development risk in frequent requirement changes.Acknowledgment

The research was supported by Ministry of Science and Technology research project funds (Project No.: MOST 105-2221-E-158-002)

References

[1] Eveleens J. L., & Verhoef, C. (2010). The rise and fall of the chaos report figures. *IEEE Software*, *27(1)*, 30 – 36.

[2] Boehm, B. W. (1991). Software risk management: Principles and practices. *IEEE Software*, *8(1)*, 32 – 41.

[3] Pressman, R. S. (2010). *Software Engineering: A Practitioner's Approach*. New York: McGraw-Hill.

[4] Schach, S. R. (2011). *Object-Oriented and Classical Software Engineering* (8th ed.), New York: McGraw-Hill.

[5] Larman C. & Basili, V. R. (2003). Iterative and incremental development: A brief history, computer. *IEEE CS Press*, *36(6)*, 47 - 56.

[6] Martin, R. C. (2002). *Agile Software Development*, *Principles, Practices and Patterns*, Prentice Hall.

[7] Martin fowler. *Continuous Integration*, martinfowler.com. Retrieved May 1, 2006, http://www.martinfowler.com/articles/continuousIntegration.html

[8] Duvall, P. *Continuous Integration Servers and Tools*. Retrieved May 1, 2015, from https://dzone.com/refcardz/continuous-integration-servers#

[9] Duvall, P., Matyas, S., & Glover, A. (2007). Continuous integration: Improving software quality and reducing risk.

[10] Saff, D., & Erns, M. D. (2003). Reducing wasted development time via continuous testing. *Proceedings of the IEEE International Symposium on Software Reliability Engineering*.

[11] Hornstein, H. A. (2015). The integration of project management and organizational change management is now a necessity. *International Journal of Project Management*, *33(2)*, 291 – 298.

[12] Savolainen, P., Ahonen, J. J., & Richardson, I. (2012). Software development project success and failure from the supplier's perspective: A systematic literature review. *International Journal of Project Management*, *30(4)*, 291 – 298.

[13] Stewart J. (2015). Top 10 reasons why projects fail. (2015). Retrieved May 1, 2015, from http://project-management.com/top-10-reasons-why-projects-fail/

[14] Symonds M. (2011). 15 causes of project failure. Retrieved from: https://www.projectsmart.co.uk/15-causes-of-project-failure.php

[15] Szalvay, V. (2004). An introduction to agile software development. *Danube Technologies*, 1-9.

[16] Booch, G. (1994). Object-Oriented Analysis and Design with applications. *Addison Wesley Longman*.

[17] Beck, K. (2003). *Test-Driven Development: By Example*. Addison-Wesley.

[18] Beck, K. (2004). *Extreme Programming Explained: Embrace Change*, Addison-Wesley.

[19] Fenton, N. E. (1991). *Software Metrics - A Rigorous Approach*, London: Chapman & Hall.

[20] Galin, D. (2004). Software Quality Assurance – From theory to implementation. *Pearson Education Limited*, England.

[21] Boehm, B. W. (1981). *Software Engineering Economics*, New Jersey: Prentice-Hall.

Sen-Tarng Lai was born in Taiwan in 1959. He received his BS from Soochow University, Taiwan in 1982, master from National Chiao Tung University, Taiwan in 1984 and PhD from National Taiwan University of Science and Technology, Taiwan in 1997. His research interests include software security, software project management, and software quality. He is currently an assistant professor in the Department of Information Technology and Management at Shin Chien University, Taipei, Taiwan.