APPLYING CONTINUOUS INTEGRATION FOR INCREASING THE MAINTENANCE QUALITY AND EFFICIENCY OF WEB APP

Sen-Tarng Lai

Dept. of Information Technology and Management, Shih Chien University,
Taipei, 10462, Taiwan

ABSTRACT

In order to project resource management and time control, software system needs to be decomposed into subsystems, functional modules and basis components. Finally, all tested components have to integrate to be the complete system. Applying IID (Iterative Incremental Development) mechanism, agile development model becomes the practical method to reduce software project failure rate. Continuous integration (CI) is an IID implementation concept which can effectively reduce software development risk. Web app with high change characteristic is suitable to use agile development model as the development and maintenance methodology. The paper depth surveys CI operating environment and advantages. Introducing CI concept can make up the moving target problems to impact of Web app. For this, the paper proposes a Continuous Integration based Web Applications Maintenance Procedure (CIWAMP) to assist the system integration operating. Based on CI characteristics, CIWAMP makes Web app can be deployed quickly, increase stakeholder communication frequency, improve staff morale, and effectively reduce Web app maintenance quality and efficiency.

KEYWORDS

Continuous Integration, agile process, Web app, integration test, maintenance quality and efficiency

1. INTRODUCTION

Web applications (Web app) development process must overcome the challenges of environmental changes and many kind of services requests. For effectively reducing development risks, Web app must decrease the impact of all of changes [1], [2]. Many events, which include error correction, requirement specification revision, environment evolution, and resource adjustment often cause to project plan changes. Any plan change always affects software development operation. Plan changes not only have to invest extra resource and cost, but also cause some related bugs and issues [3], [4]. In requests change process, the affected development documents and source code unable to effectively revise, test and integrate, will increase Web app development risk. In addition, the implied issues and produced bugs unable to identify quickly and modify timely. The issues and bugs will flow to the follow phases to greatly reduce Web app success ratio. Many factors may affect Web app development risk. However, one of critical factors is the bugs and issues cannot immediately identify and revise. For this, Web app development must enhance the defects identification and quick revision mechanism to reduce risks of environmental changes and modification requests.

In current software development process, Iterative and Incremental Development (IID) mechanism has become a new development trend [5], [6]. RUP (Rational Unified Process) is a widely used development model of object-oriented analysis and design [6]. Agile process very

cares workable software and uses time-boxing approach to control time and items change risks [6], [7], [8]. Using agile development model, the small scale software project has higher success ratio. RUP and agile process combine with the IID mechanism to implement the software development work. Iterative development is to develop job to repeatedly use the same series of actions or steps, incremental development means that every time will be in the existing software systems, increase and other new features, modules or units. Agile development model to develop after each iterative operation complete operating system version, and immediately test and evaluate this version. The processes problems and the product defects can be easily detected to help timely modify and correct measures to reduce the software development risks. IID mechanism gradually be taken seriously, many software methodologies combine with IID to improve development quality and productivity, among which continuous integration (CI) with agile development model most cited and discussed [9], [10], [11], [12], [13]. CI is a concept of implementation. CI implementation immediate advantage can early find software system problems and defects, reduce software development risks.

Web app with high change characteristic is suitable to use agile development model as a development and maintenance methodology. In Web app development process, agile model with CI development concept can effectively find the problems and the defects early, and timely correct to avoid the defects outspread, greatly reduce Web app development risks. Based on CI features, the critical components of CI environment should be clearly defined and combined to set up a perfect CI operating environment. In addition, for building the common recognition of software developers, the paper proposes the Continuous Integration based Web Applications Maintenance Procedure (CIWAMP). Using CI environment for Web app maintenance, automated integration testing can quickly assist product deployment, improve stakeholder communication, increase staff morale and other characteristics, and effectively increase quality and productivity of Web app maintenance. In Section 2, the advantage of agile development model and the CI features are discussed. In Section 3, describes the characteristics of Web app and discusses the existed maintenance problems of Web app. CI concept must with a sound operating environment to play to it strengths, in Section 4, defines the perfect CI environment and presents the CIWAMP. In Section 5, evaluates the advantages of CI with CIWAMP. In Section 6, emphasizes the strengths of CIWAMP to increase the maintenance quality and efficiency of Web app, and to draw conclusions for the topic.

2. AGILE DEVELOPMENT AND CONTINUOUS INTEGRATION

Combining agile development model with CI can increase software development quality and productivity.

2.1. AGILE SOFTWARE DEVELOPMENT

In recent twenty years, the growth of information technology, operational environment and user requirements, software development methodologies continuously progress. Most software development models have high relationship with the user requirements. Water fall model defines clear phase mission and very concerns the phase documents [6]. 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 [5], [6]. By iterative development model, user can

incrementally provide the requirement items. Software development risks can be greatly reduced. Each development model should has the adjustment strategy to handle the change of user

equirements. The adjustment strategy can't effectively reduce the software development risks that may impact success ratio of software project.

In February 2001, seventeen software developers met at the Utah (ski resort) of USA and produced the Manifesto of the agile software development [6]. Many of participants had previously authored their own software development methodologies, including Extreme programming, Crystal, and Scrum [6], [7], [8]. Agile software development proposes several critical viewpoints [7], [8]:

(1) In development process, does not concern analysis and design phase operations and documents.

(2) As soon as possible to enter programming phase, workable software is more practical than development document.

(3) Support and provide high changeability software system.

(4) Enhance the cooperative relationship between developer and user, user can fully attend the development team.

In time management side, agile software development applies time-boxing approach to control process schedule [6], [8]. Requested software project must release a new version in two or three weeks. Let client clearly understand the development progress and test, audit the requirement of new version. In each day, a fifteen minutes stand up meeting is fixedly held to effectively reach the fully communication between client and developers. In addition, agile process uses iterative and incremental development to reduce the requirement complexity, refactoring to increase the requirement modified flexibility, non-document oriented can reduce the cost of requirements change. In development change impacts, agile development greatly decreases schedule delay, cost excess budget and quality unsatisfied user requirement situations, software development change risk can be effectively reduced. However, agile process neglects the formal analysis and design phases, uses non-document oriented development, and doesn't pay attention to follow-up maintenance operation that are major drawbacks. For effectively reducing the software development risks, the drawbacks of agile process should be concretely improved or enhanced.

2.2. ADVANTAGES OF CONTINUOUS INTEGRATION

CI is a software engineering practice in which new items and changes are immediately tested and reported on before they are check-in to a source repository. The goal of CI is to identify the software errors and defects quickly, and to make the errors and defects can be corrected and revised as soon as possible. In 1994, Grady Booch mentioned the CI phase in his book of Object-Oriented Analysis and Design with Applications [14]. In 1997, Kent Beck and Ron Jeffries proposed XP which included CI concept [9], [11]. In 1998, Kent published a paper of CI which valued and supported the importance of directly communication technology. Based on IID mechanism, agile development combines XP [15], [16], TDD (Test Driven Development) [17], and BDD (Behaviors Driven Development) methods [18], and makes CI to become a main stream and important trend of software development. Focus on the automatic and repeat process are the major characteristics of CI. Therefore, integrated with related automated development tools is first mission of CI environment.

In software development process, software project management applies modularization to enable teamwork development and make complex system manageable. Modularization decomposes the complex software system into many manageable program units [6], [19]. Each program unit just handle a specific function, therefore the program units have to work together and integrate into a module function, subsystem and system. CI through more times integration and testing makes the

errors and defects can be identified quickly [20]. Errors correction and defects revision can be handled timely and greatly avoid to transfer and extend to the follow phases. Summarize five major advantages of CI and discuss as follows (shown as figure 1):

- High efficiency: CI combines many automated tools and procedures can increase the development and operation efficiency.
- Reduce development risks: According to development requests, CI can more times integrate in a day to reduce software project development risks.
- Smooth communication: CM system, source repository and workable software can assist build common understanding among stakeholder.
- Improvement quality: CI follows the systematic procedures and combines useful automated tools to improve software development quality.
- Increased morale: CI increases communication frequency, dramatically reducing communication barriers and software development risks. The development team confidence can be established and morale can be increased.

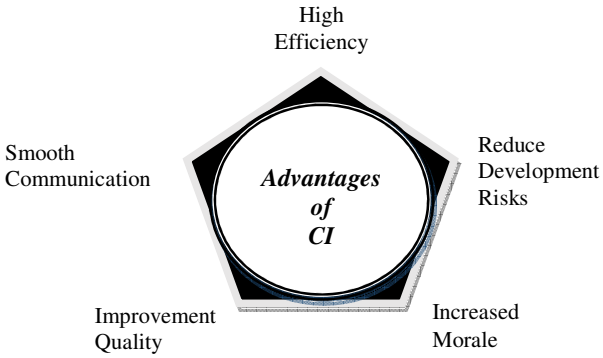

Figure 1. Five major advantages of CI

3. WEB APP CHARACTERISTICS AND MAINTENANCE PROBLEMS

High changeability is a critical factor to cause the Web app development risks. Quickly handle change requests can reduce change risks of Web app development.

3.1 CHARACTERISTICS OF WEB APP

In Web app development process, it is necessary to face challenge of many changes. Updating, requirement specifications revision, environment evolution, and resource adjustment etc. situations possibly cause to fail of Web app development. It is because that some characteristics of Web app differs from the general applications. About the characteristics of Web app are described as follows:

- Friendly user interface: In general, Web app user can't accept complex interface. Simplicity and high friendly user interface is a critical factor to attract user and also is a first step of Web app success.

- Bear a wide range of users: Web app belongs to public browsing system. Therefore, it must have high tolerance to accept all kinds of user and accept many kinds change or function requests

- Rapidly accomplish change service requests: With rapidly changing technology and a wide range of Internet users, multi-styles change service requests are submitted continuously. Web app with high maintainability can quickly accomplish change service requests.

- Timely add new service items: Belong to the public browsing system, Web app need deal with many views and new service requests. The proposed service requests by related units have its timeliness. Therefore, high maintainability is necessary condition to timely add new service items of Web app.

- Quick migration to new platform: Subscriber of successful Web app will grow continuously. For providing high service quality, operating environment of Web app should be timely adjusted or enhanced. Web app can timely migrate to the more perfect working environment, app life cycle can be extended concretely.

- High integration ability: Multi-layers architecture operation environment is a key attribute of Web app. In order to provide multi-style services, Web app should emphasize on integration and combination with the related systems. High integration ability is a necessary and critical characteristic of Web app.

Multi-layer architecture is the major operating environment of Web app. Therefore, Web app development very care about analysis, preliminary design and detailed design. Analysis and preliminary design focus on system architecture design and external entities interface design which belong to system planning and external interface related design. Detailed design focuses on the modularized function design and basic level component design which belong to system internal data structure and logic operation related design. According to operating environment, types of user, development tools and maintenance manner, special attributes of Web app can clearly identify the difference with general application software.

3.2 WEB APP MAINTENANCE PROBLEMS

For meeting Web app characteristics, Web app development should use modular design to increase changeability, maintainability, friendly user interface and connectivity of external entities. Based on the preliminary and detailed modular design, Web app maintenance issues can be distributed to four stages that describe as follows:

- Unit testing problem: Change frequency and schedule pressure make unit testing often be omitted or disregarded. Incomplete unit testing always causes problems and defects and flow to the follow phases. For reducing unit testing risk, the unit testing procedure should be clearly defined and automated testing tools need be suitably applied [16], [20].

- System integration problem: based on software system combined relation, related unit programs are integrated into the function modules, subsystems and system. Lacking automated tools and procedures, integration process often be delayed or not passed that causes errors and defects flow to the follow development phases.

- Communication problem: communication barriers between the user and developer is the key factor to impact the success. Demo workable software can increase common recognition between the user and developers [6], [7], [8]. In change requests environment, Web app

should meet user requests and timely release new workable version for the users to increase trust ability. However, if Web app cannot timely release new workable version, then communication barriers may expand and cause the development risk.

- Quality and efficiency problem: In Web app development and maintenance process, there are many internal modules and external entities must be integrated. Interface mistakes always cause system problems and defects which deeply impact the quality of software application. For improving application quality, the development time must be extended and development budget must be added. Internal and external interface design of Web app is the other key development risk.

Integration testing operations are assigned to the testing phase and not enough test frequency. The frequency, time and automated tools of integration testing are critical reason of Web app maintenance problems. It is because the problems and defects cannot be discovery immediately and cause the errors extension to the follow maintenance operation.

4. CI DEVELOPMENT ENVIRONMENT

For achieving the features of CI, a CI environment must be build and the CI operating procedure should be set up.

4.1 High quality class component

In order to effectively improve the operational efficiency and quality of CI, the definition and design of class components should have the following quality characteristics:

(1) Independence and modularity:

- Appropriate inheritance: The component inherits the parent class resource appropriately, and the effect of reuse can be achieved. However, too many levels of inheritance will cause the component to lose independence and modularity.
- High cohesion: Class components are designed to be functional or information cohesion, which can improve the modularity of the class components.
- Low coupling: The coupling of data or data structures can improve the modularity of class components.

(2) TDD-based unit testing:

- Complete test cases: Before the detail class design, the complete test case must be designed according to the class component specifications [17].
- Combining with unit testing tool: Unit testing must be able to take automated testing for unit testing [22].
- Assistant integration testing: A complete and clear interface design that can help with ongoing integration efforts.

(3) Maintainability:

- Changeability: In line with the continuous evolution of the environment and the constant changes in requirements, class components must have features that are easy to modify and adjust.

- Extensibility: In order to improve the efficiency of maintenance, the class components should have the ability to continuously expand to increase the efficiency of maintenance operation.

(4) Interface integration ability:

- Internal data integration: In the process of continuous integration, in order to compliance the operational process and efficiency, the class components should refer to the existing internal data structure in advance, and design an interface that can be integrated with the internal data structure.

- External data integration: In the continuous expansion and integration process, in order to compliance the external entity operation mode and data format, the class components should refer to the existing external entity data structure in advance. And timely design the integration interface with the external entity data structure.

Using LCM (Linear Combination Model) [23], [24], relevant quality factors of class component can be incorporated into quality metrics, then related quality metrics can be incorporated into quality measurements, and finally, quality indicator for class component can be generated. The quantified quality measurement combination process is called a Class Component Quality Measurement (CCQM) Model (shown as figure 2). Using CCQM can timely identify the quality defects of class component, and concretely improve the quality of class component.

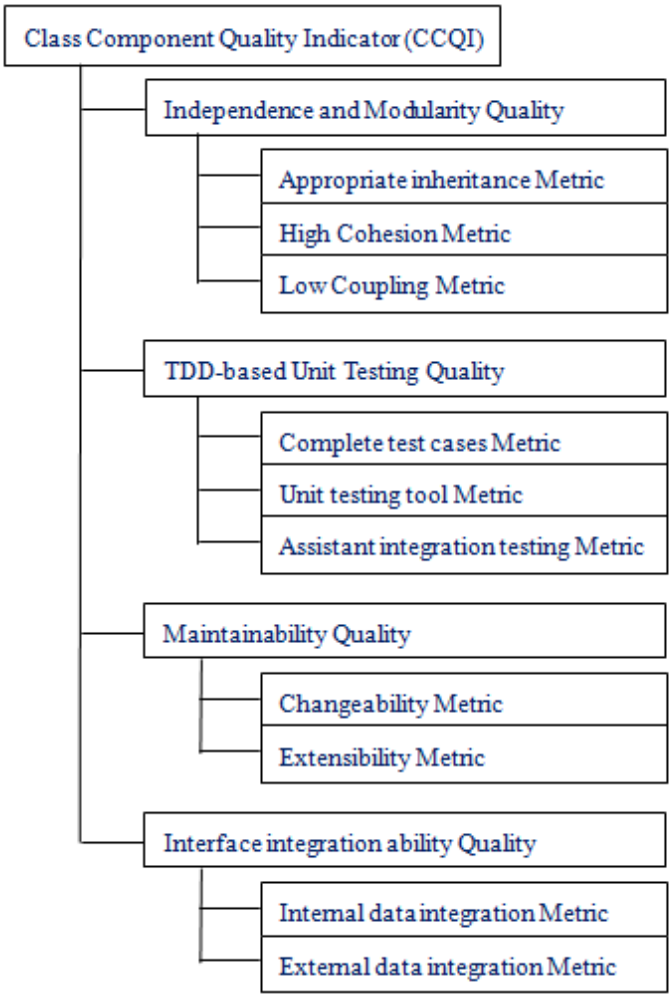

Figure 2. Architecture of CCQM model

4.2 MAJOR FEATURES OF CI ENVIRONMENT

In order to quickly announce the workable software, software development environment must combine high efficiency automated tools and development procedures. CI environment should connect with six major features for achieving the workable software of high quality and high efficiency. The major components describe as follows (shown as figure 3):

(1) Automated testing tools: Generally, task of unit testing belongs to the responsibility of programmer. After coding, unit program needs proceed complete unit testing. Pass the test qualified criteria, unit program can check-in to resource repository for managing by CM system. For improve unit testing quality and productivity, CI environment should plan the suitable and automated unit testing tools (ex. xUnit, jUnit) for assisting unit testing operations of the programmer [22].

(2) CM (Configure Management) procedure and version control: In order to management software system various versions, version control is necessary tool to recovery the previous versions and identify the version difference. In order to monitor and control software

changes, CM procedure is a necessary system to record the revision reason, date, programmer and detailed contents. In order to continuously control and manage software system versions, CM procedure and version control tools (ex. git, csv) [25] are the necessary component of CI environment.

(3) Source repository: A unity repository is an important communication channel for development team. The development documents, source code and related documents should save to source repository to share by the stakeholder. However, access security of source repository must be carefully controlled and managed.

(4) Automated integration: The modified, extended or updated source codes need be recompiled, retested, and integrated into function modules. Based on saved time of file, module integration script language can identify the new added or revised source codes and enforce compiling and linking into function module operations. According to the integration procedure, at first, module integration testing need be executed and pass verification, and then subsystem integration testing need be accomplished and pass verification. Finally, based on system structure, all subsystems are integrated into a system and has to be accomplished testing and verification.

(5) Automated deployment: After system integration test, automated deployment is a next important step to quickly install new working software on user site. Automated deployment can increase software development efficiency and identify new workable software errors and defects timely.

(6) Problems and defects detector: identify the software system problems and defects as early as possible, the revision work can be simplified, modification effort can be reduced. New version software accomplished deploying, the software need proceed complete internal verification and user validation. Based on the detection procedure, verification and validation may identify the problems and defects of new version software. Collect, analyse and record the problems and defects, the correction and revision operations should be planned and executed as quickly as possible.

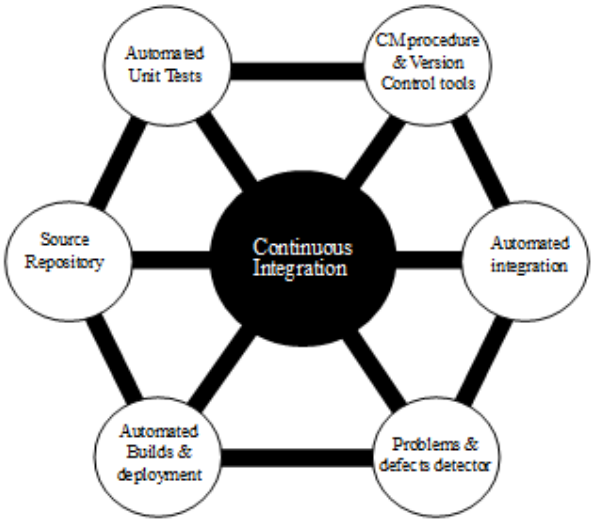

Figure 3. CI Environment

4.3 CONTINUOUS INTEGRATION BASED WEB APPLICATIONS MAINTENANCE PROCEDURE

CI environment combines useful automated tools and necessary development procedures for increasing software development quality and productivity. In addition, CI needs to plan a complete operation procedure to require compliance the development process. In this paper, based on the Web app development, combining agile development and CI to propose the Continuous Integration based Web applications Maintenance Procedure (CIWAMP) (shown as figure 4). CIWAMP divides into five phases that describe as follows:

(1) Unit testing phase: Based on TDD, the qualified user stories can derived to the complete test cases [17], [21]. The test cases and unit test tool can assist automated unit testing. The unit programs, which passed the unit testing criteria, can enter next phase, otherwise the errors of program must be identification and debugging. Until unit testing result can pass testing criteria.

(2) CM phase: Passed unit testing requests, unit programs have to check-in source repository to manage by CM system. According to CM procedure, unit program and test cases need check-in and save to source repository. In order to keep more detailed activity information, program developer, date time, reason, and program relationship also should be saved into the CM system. Enter the integration and deployment phase, related programs need check-out from source repository. According to module, subsystem and system integration script language, related unit programs need check-out and get from source repository for assisting the follow development tasks.

(3) Integration and deployment phase: In preliminary design, multi-layer products should be complete planned and detail defined. Especially, the architecture of function modules, subsystem and system have to refer the formal script language to define combined relation and construction steps. Building definition file is an important document to assistant generate function module, subsystem and system. Based on the version date time of source code, object code and image file, building definition file can automated recompile and rebuild the target products. Function module, subsystem and system building script files describe as follows:

- Function module building script: based on function module combined relation, the unit programs and corresponding object codes are defined, and specific describe compiling, linking and loading styles and sequence.

- Subsystem building script: based on subsystem combined relation, the function modules and corresponding object codes are defined, and specific describe compiling, linking and loading styles and sequence.

- System building scrip: based on system combined relation, the system and corresponding images are defined, and specific describe compiling, linking and loading styles and sequence.

(4) Verification and validation: Repeated integrating in CI environment, new version workable software can be quickly created. Therefore, based on the workable software, the user can attend and assist software verification and validation. The user's mind try to assist in identifying problems and defects of a new version, and respond immediately to the development team. Establish good channels of communication between stakeholders through validation and verification system. Developers determine the problems scope and

defects causes that may occur. Operation job should return to the corresponding development phases to handle problems modification and defects correction, and prepare for next integrating tasks [26].

(5) Continuous improvement phase: When user stories (requirement items) already passed the system validation and verification, the software development has been completed and entered the stage of maintenance work. This phase is still ongoing CI operations, and the number of times integrating more frequently. In order to improve the problems and defects of CIWAMP in CI operation process. Relevant data should continually collect to identify problems and defects of CI components, procedures and operating environment for continuously improving the CIWAMP.

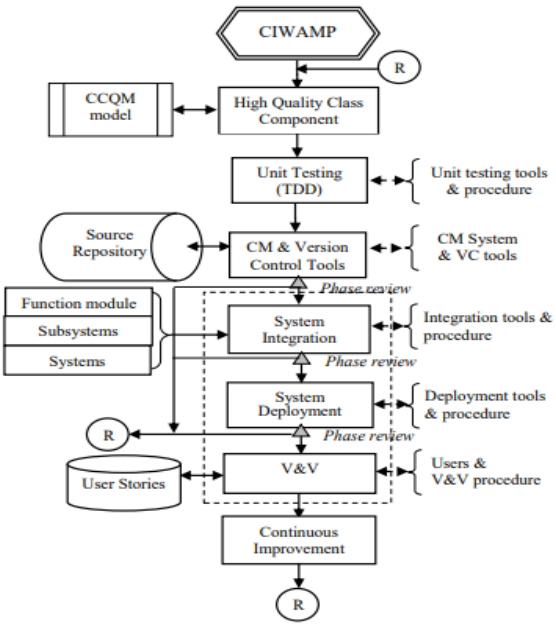

Figure 4. Operation flowchart of CIWAMP

5. EFFICIENCY EVALUATION OF CIWAMP

CIWAMP combines the characteristics of agile software development, high quality class components, and the advantages of CI. CIWAMP uses the suitable automated tools and development procedures to efficiently improve Web app quality and productivity. Evaluation CIWAMP five advantages described as follows: (shown as Table 1):

(1) Errors and defects identification: traditional software development model handles integration testing only in test phase and test a limited number of times. In agile development, CI with CIWAMP adopts automated tools which can timely handle unit testing, integration testing and deployment tasks. More testing can more quickly identify program errors and software defects.

(2) Communication channel and developers morale: communication barriers between the user and developer is the key factor to impact the success. CI with CIWAMP uses version control tool, the source repository and workable software to establish the interactive and open communication channels. CI increases communication frequency, dramatically reducing communication barriers and software development risks.

(3) Automation Ratio: CI environment applies many automated tools for TDD-based unit testing [17], regression testing, CM, version control and integration testing to increase maintenance quality and efficiency. Traditional methods use also CASE tools for increasing maintenance operation efficiency. However, the traditional automation tools don't overall consideration for IID process. Therefore, the automation ratio can partly complete.

(4) Early correction and revision: generally, in the testing phase, traditional development model correcting program errors and modify software defects. It often causes the errors and defects transmission and extension to the follow phases. CI with CIWAMP specially concerns the errors and defects identification in early phase. Errors and defects cannot be timely corrected or removed that may transfer to the follow phases and cause software development with high risk.

(5) Quality and productivity: CI environment combines many automated tools can reduce personnel intervention and increase software quality. Integration procedures can quickly construct the basic components into the functional modules, subsystems and system. CI with CIWAMP not only can improve development efficiency but also can increase software quality.

For evaluating the advantages of CIWAMP, five impact items include defects or probloems indentification, communication capability and developer morale, automation ratio, early correction and revision, and quality and productivity etc. to evaluate the CIWAMP (shown as Table 1).

Table 1. CIWAMP evaluation table

Advantages of CIWAMP	Traditional Procedure	CIWAMP
Defects or probloems indentification	Later	Early
Communication capability and Developer morale	Weak	Strong
Automation Ratio	Partly	High
Early correction and revision	Weak	Strong
Quality and Productivity	Partly	High

6. CONCLUSION

Web app is a key information system to accomplish the high quality and high efficiency transaction activities. Web app must quickly overcome environment change, and have the ability to continuously adjust and modify for meeting the diversified needs of the users. Therefore, how to increase the maintenance quality and efficiency of Web app is a topic worthy of further exploration. In this paper, combining the characteristics of agile software development, high quality class components, and the advantages of CI, proposes the CIWAMP. CIWAMP divided into five phases which include automated unit testing, configuration management and version control, automated integration and deployment, verification and validation, and continuous improvement. In each phase, CIWAMP uses the suitable automated tools and development procedures to efficiently improve Web app quality and productivity. In Web app maintenance process, applied CIWAMP may reach four advantages as follows:

- Users can quickly try the updated version
- Build the confidence of Web app stakeholder and users
- Problems and defects of Web app can be detected earlier, and immediate improvements
- Increase the maintenance quality and efficiency of Web app

Acknowledgments.

This research was supported by the Shih Chien University 2018 research project funds (Project No.: 107-08-01001)

REFERENCES

[1] Brandon, D. M. (Ed.). Software Engineering for Modern Web Applications: Methodologies and Technologies, IGI Global, 2008.

[2] Al-Fedaghi, S., "Developing Web Applications," International Journal of Software Engineering and Its Applications, Vol. 5 No. 2, April, 2011, pp.57-68.

[3] Boehm, B.W., "Software risk management: Principles and practices," IEEE Software, vol. 8, no.1, 1991, pp.32-41.

[4] Fairley, R., "Risk management for Software Projects," IEEE Software, vol. 11, no. 3, 1994, pp. 57-67.

[5] Larman, C. and Basili, V. R., "Iterative and Incremental Development: A Brief History", Computer, IEEE CS Press, 2004, pp. 48.

[6] Schach, S. R., Object-Oriented and Classical Software Engineering, Eighth Edition, McGraw-Hill, New York, 2011.

[7] Robert C. Martin, Agile Software Development, Principles, Practices and Patterns, Prentice Hall, 2002.

[8] Szalvay, V., An Introduction to Agile Software Development, CollabNet, Inc., 2004.

[9] Sthl, D., Mrtensson, T., & Bosch, J., The continuity of continuous integration. Journal of Systems and Software, 127(C), 2017, 150-167.

[10] Shahin, M., Babar, M. A., & Zhu, L., Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. IEEE Access, 5, 2017, 3909-3943.

[11] Fowler, Martin, "Continuous Integration," martinfowler.com, http://www.martinfowler.com/articles/continuousIntegration.html (1 May 2006).accessed Nov. 9, 2018

[12] Duvall, Paul, Continuous Integration Servers and Tools, DZone Refcardz. https://dzone.com/refcardz/continuous-integration-servers#, (accessed Nov. 11, 2018)

[13] Duvall, Paul, Matyas, Steve and Glover, Andrew, Continuous Integration: Improving Software Quality and Reducing Risk, Pearson Education, Inc., 2007.

[14] Booch, Grady, Object-Oriented Analysis and Design with applications 2nd edition, Addison Wesley Longman1994.

[15] Beck, K. "Extreme programming: A humanistic discipline of software development," Fundamental Approaches to Software Engineering, 2006, pp. 1-6,

[16] Crispin, Lisa and House, Tip, "Testing Extreme Programming", Addison Wesley, 2003.

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

[18] North, Dan, "Introducing BDD," http://dannorth.net/introducing-bdd/ (accessed Nov. 9, 2018)

[19] Bavota, G., et al. "Using structural and semantic measures to improve software modularization," Empirical Software Engineering vol. 18 no. 5, 2013, pp.901-932.

[20] Saff D. and Erns, M. D., "Reducing Wasted Development Time via Continuous Testing," Proceeding of IEEE International Symposium on Software Reliability Engineering (ISSRE), 2003, pp.281-292.

[21] Wells, Don "Code the Unit Test First", http://www.extremeprogramming.org/rules/testfirst.html (accessed Nov. 9, 2018)

[22] Cheon, Y. and Leavens, G. T., A simple and practical approach to unit testing: The JML and JUnit way. In European Conference on Object-Oriented Programming, Springer, Berlin, Heidelberg, 2002. pp. 231-255

[23] Fenton, N. E., Software Metrics - A Rigorous Approach, Chapman & Hall, 1991.

[24] Galin, D., Software Quality Assurance – From theory to implementation, Pearson Education Limited, England, 2004.

[25] Loeliger, J., and McCullough M., Version Control with Git: Powerful tools and techniques for collaborative software development, O'Reilly Media, Inc., 2012.

[26] Fowler, Martin, "Refactoring Improving The Design Of Existing Code," Addison-Wesley, 1999.

AUTHOR

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.