Deployment and Evaluation of a Continues Integration Process in Agile Development

Hsin-Ke Lu, Peng-Chun Lin, Pin-Chia Huang, and An Yuan
Department of Information Management, Chinese Culture University, Taipei, Taiwan
Email: {sklu, pclin, pchuang, ayuan}@sce.pccu.edu.tw

***Abstract*—In addition to the strong market demands, software development projects are facing constantly change of requirements from dynamic business context. Striving to meet the project deadline, in budget and maintain quality of systems brings up the importance of development automation. The benefits of continuous integration of agile development have been reported in the academic literatures, however, the consideration on determining the necessary procedures during the implementation and how to quantify the performance and effectiveness on the implementation is not widely discussed. An automation continues integration process was developed in this study. For evaluating on the performance and acceptance of the process implemented, a new evaluation framework was constructed as measurement tool. The analysis and discussions of research result will serve as a reference for the rear researches and for the system development practice on transitioning to agile development.**

***Index Terms*—agile development, continuous integration and delivery**

I. Introduction

Software application demands are increasing rapidly, in order to meet market needs and deliver quickly to stay competitive, new features and capabilities are required to build constantly, high expectations are being placed on the IT department, to be able to deliver on time and without compromise system quality becomes the critical issue.

Standish Group [1] reported that on more than 50,000 software projects in the United States, it shows 31.1% of projects are cancelled before they ever get completed, 52.7% of which will cost 189% more than their original estimate budgets, there are only 16.2% of software projects that are completed on time and on budget, and the numbers are lowered to 9% for larger companies. Even with projects that are completed, only approximately 42% of the originally proposed features and functions are deployed. With such staggering numbers presented, it is obvious that something must be wrong in the line of software development process. The two most important factors causing software project to fail or challenged was incomplete requirements and lack of user involvement, see Table I.

TABLE I. Project Challenged Factors

	Project Challenged Factors	Responses
1	Lack of User Input	12.8%
2	Incomplete Requirements & Specifications	12.3%
3	Changing Requirements & Specifications	11.8%
4	Lack of Executive Support	7.5%
5	Technology Incompetence	7.0%
6	Lack of Resources	6.4%
7	Unrealistic Expectations	5.9%
8	Unclear Objectives	5.3%
9	Unrealistic Time Frames	4.3%
10	New Technology	3.7%

Over the years, enterprises tried to adopt different processes to optimize their software development practices, but the focus has been mainly on software development, leaving the operations side of software delivery lagging behind. Now a day IT practitioners often criticize the traditional waterfall-based development approach on its linear and structured inwardness. This model mainly relies on a predictable environment and progress to be able to succeed, but the reality is that the environment and requirements are constantly changing and is often unpredictable [2]. Agile development was proposed as one of solutions to solve that by iterating the development process in short intervals and incrementally enhance the functionality to meet the requirements. However, this process lacks the consideration of how the adaptive changes affect the performance and integration of the project.

Successful software project requires comprehensive project management, agile developing practices, orchestrated quality control and align with organization culture, communication and collaboration with stakeholders. In order to achieve this, as soon as development done, the delivery process is expected to integrate automatically to the operational level [3]. With continuous integration, the system automatically builds artifacts and checks for errors. The whole releasing process should include automated planned and orchestrated tests to insure the stable and available. To discover defects before every build is the way to deliver reliable and predictable releases [4].

The continuous integration and delivery of agile development is regarded as one of solutions to these tough issues of system development. The purpose of this study is to propose an automation approach of continuous integration, construct an evaluation framework for the new process implemented and the analysis can be served

Manuscript received September 15, 2017; revised November 24, 2017.

doi: 10.12720/jait.8.4.204-209

as a reference for rear academic researches and practitioners for deploying agile development approach.

II. Theoretical Background

A. Agile Development

The traditional waterfall development methodology is inflexible with its linear design and that it relies on theories that assume the environment and requirements are predictable. However, the environment and requirements are constantly changing and often unpredictable [2]. Agile method was proposed for these issues, which basically put together as a solution to circumvent to pitfalls associated with waterfall model. It is to have close collaboration between the development team and business stakeholders to achieve a more frequent delivery to meet the business market needs [5], [6].

Agility is becoming more important due to the dynamic market demands and the pace of technology change [7]. As agile development becomes more popular, organizations should adapt the new fundamental nature of software development. It reflects the need to adapt to variations in requirements, resources, and uncertainty [8].

B. Continuous Integration and Delivery

Continuous Integration (CI) is a software development practice where systems (subsystems) are integrated frequently during the development phase to avoid unpredictability and expensive integration effort caused by the separated integration phase [9]. For implementing CI method, the development team integrates changes daily and that each integration is verified and build automatically [10]. With the increased in popularity of agile development, CI has become an essential ingredient for teams doing iterative and incremental software delivery, with the combination of agile development with XP, Test-Driven Development (TDD) and Behaviors Driven Development (BDD), CI has become an important trend of software development [11].

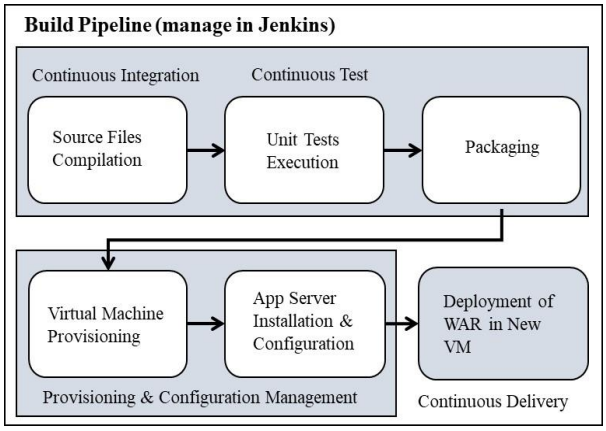

Figure 1. Continuous integration and continuous delivery.

Following closely on CI, Continuous Delivery (CD) is a series of practices used in software development process where software can be rapidly and reliably release to an environment at any time. Getting software released to the market is often an inefficient, risky, and time-consuming process, the release cycle often takes months or even years for large enterprises. Continuous delivery can help optimize this process through a more efficient, reliable and low-risk releases and continuously adapt software in line with user feedbacks, shifts in the market and changes to business strategy [12].

Soni [13] articulated that the new approach is focused on automation tools to achieve end to end automation, and it shows that by adopting different type of tools in the technology stacks, automation approaches maybe different in terms of pull or push mechanisms of artifacts from continuous integration to continuous delivery, see Fig. 1.

III. Deploying a Process of Continues Integration

To achieve the speed and agility, it is imperative to choose the right tools to enable automation across all aspects of development, production, and operations. Puppet Labs [14] reported that more than 80% of high performing software project organizations rely on automated tools for management and deployment. In this study, a case which deployed the new process of continues integration in software development projects was illustrated and analyzed. The implementation was adopted on the Java project development team. The team had begun the adoption of the automation continues integration process in mid-2015, the main goal was to increase productivity and reduce the time spent caused by fixing unexpected code defect or failed deployment. Among the desired improvements were to reduce code verification feedback time and increase efficiency on communication and collaboration with other stakeholders. Before the model adoption the team begun the practice of agile methodology using Scrum 6 months earlier, there were daily standup meetings to pinpoint the major challenges during software development. The adoption of the automation model is expected to further reduce the challenges the team face during the development phase and to improve the efficiency of the development process.

When the Scrum process is adopted, JIRA was chosen to track issues and project progress, as well as status management. It is also used as a communication and collaboration tool with other stakeholders. The server implemented was Jenkins CI server, it is an open source automation server written in Java. It acts as the master build server for automatic integration, CI tasks are configured to pull the latest code from the subversion server and automatically build the codes with Ant at a specific time daily.

This automation process of CI is shown in Fig. 2. Each software project runs its own CI pipeline to provide feedback for the developers working with that particular project, the pipelines run in parallel as there are multiple projects. If the build fails, the CI server sends out messages to notify the developers. For integration and acceptance testing, the software is packaged, run the regression test, and installed to separate test environments automatically as part of the CI process, the packaged

application will then be pushed to a remote file server and tagged for client side deployment.

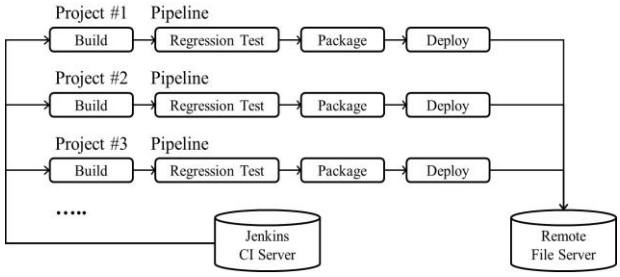

Figure 2. Automated integration process.

When the build is success the regression test will be run against all unit-test classes, again it should pass without error or the test will be considered failed and the build will be reverted. Once the build and test run is completed, a code coverage report will be available on the CI server, the report can be drill down to individual classes, and the coverage can be down to the method level, it shows the percentage of the methods, conditional statements covered in the test, it can be served as a guide for better fine tune the code test base. The deployment phase will begin at this stage, the deployment file will then be deployed to the appropriate server depend on the pipeline they are at, if the remote server is not accessible, the deployment file will be available to download on the CI server.

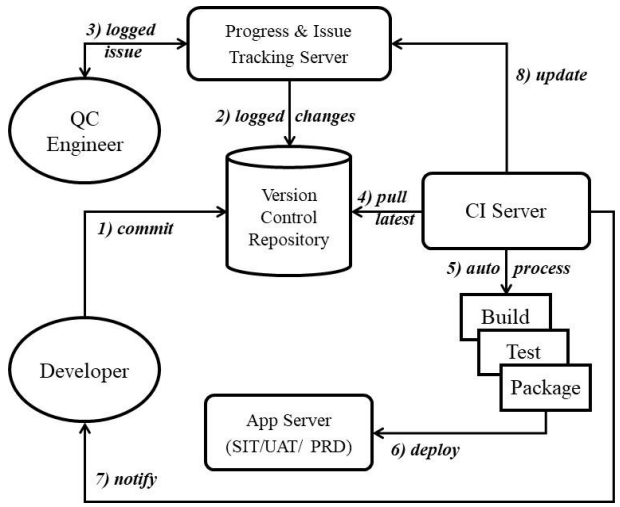

Figure 3. Automation process.

In Fig. 3, as the automation process is completed, the CI server will update the current build version status on the progress server, and the whole process just iterate until the project is complete.

The automation process ensures that the code on the repository will always be tested and in a workable state, able to discover defect code early in the process, prevent the unexpected risk with deployments, every build should be under predictable. It also eliminates the need for manual packaging and deployment process, reduce the risk on errors and cuts down the time on human labor dramatically.

IV. Evaluation Framework

The benefits of practicing continuous integration have been reported in the academic literatures in any forms [15]. The new CI automation process implemented is evaluated by a set of criteria and metric modified from an evaluation framework by Mutschler and Bumiller [16] The proposed evaluation will be performed on four criteria each with a distinct view on the performance and effectiveness during the constructing and implementing of the automation process, shown in Fig. 4.

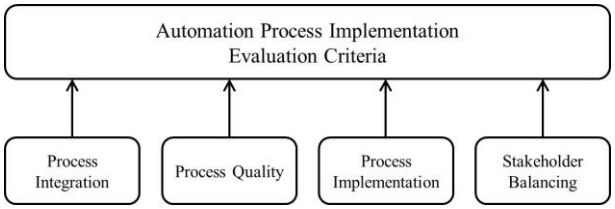

Figure 4. Process implementation evaluation criteria.

The four criteria to be evaluated are process integration, process quality, process implementation and stakeholder balancing, each criterion is built with metrics for detail measurements, which are described as follows:

Process Integration: Process integration focuses on the process level during the implementation process. The challenges of integrating new process is the convergence of the new process and how it aligns with the legacy process, enterprise cannot afford the cost and the risk to replace a business process, therefore it is important to know how a new process integration performs and the quality it brings. Metrics to quantify process integration can be the convergence time to the new process, and the quality of the integration process, which can be evaluated by the time-taken to adapt the new process, does the new process disrupt and original process and the understanding and collaboration of the new process with other stakeholders involved. This criterion can help enterprise gain better insights into the effectiveness and cost on the integration phase of the new process.

Process Quality: During a new process integration, the quality of a process will reflect the benefits of the process itself, it is critical to achieve high productivity during the development process. The quality of a new process can be identified as the improvement of the time and the quality of the implementation. Metrics to evaluate the quality of the process can be the responsiveness on the change of requirements, the improvement on delivery rate, the decrease of error rate during the development phase. This criterion can help enterprise gain knowledge on the improvement of the new process in the business context.

Process Implementation: The actual implementation of a new process could be time consuming and costly. This evaluation criterion can bring the cost consume during the implementation of a new process, which includes direct cost and hidden cost. Metrics could be quantified by the time-taken to implement or the resources invested. It includes direct cost such as time invested or the hidden cost such as the learning curve or the opportunity cost lost during the process.

Stakeholder Balancing: The stakeholder balancing should be quickly adaptable between all the stakeholders involved in the project, as the conflict and debate surfaced during the process implementation can be costly. The metrics could be quantified by the frequency of collaboration problems with stakeholders, the time to resolve the conflicts, the effectiveness on the communication with stakeholders of the new process. This criterion can bring the insights on the collaboration with other stakeholders involved during the implementation of the new process.

The questionnaire is designed based on the metrics of the criteria to show the different dimensions from the implementation process, the data will then be measured by referencing methods like software usability measurement inventory and bipolar scaling method to conduct the analysis and finalize the results. The criteria and metrics were constructed and listed in Table II.

TABLE II. EVALUATION CRITERIA AND METRICS

Criteria	Metric	Description
Process Integration	M1	The convergence time to the new process is fast
	M2	The convergence quality to the new process is high
Process Quality	M3	The new process has the benefits of improved efficiency on project execution
	M4	The new process has the benefits of improved quality on project execution
Process Implementation	M5	The time spent on the new process construction and implementation
	M6	The resources invested on the new process construction and implementation
Stakeholder Balancing	M7	The extent of conflict and debate between the stakeholders during the new process implementation
	M8	The time to create the new relationship between stakeholders after the new process implementation

There are 32 questions total in our questionnaire, the questionnaire includes a mixture of positively-keyed and negatively-keyed items; negatively-keyed items will be reverse-scored, it means that the responses will be transformed so that the item "agree" all indicate positive attitude and the undecided item will be factor out so that the attribute being measured will show the correct positive value, all the agree items will mean positive attitude towards the metric and the disagree items will be negative attitude towards the metrics.

V. ANALYSIS AND DISCUSSION

A survey is conducted using the questionnaire on the software development team to evaluate the performance and the acceptance of the new process based on the evaluation framework.

There is a total of 23 respondents, because the implementation was for Java software projects, therefore only members involved in Java projects took the survey, see Appendix C for background of the respondents. With four questions for each metric, a total of answers for each metric will be 92. The data summary of the responses based on the metrics is shown in Table III.

TABLE III. RESPONSES SUMMARY DATA

Criteria	Metrics	Agree	Undecided	Disagree
Process Integration	M1	34	38	20
	M2	33	45	14
Process Quality	M3	58	25	9
	M4	37	50	5
Process Implementation	M5	31	45	16
	M6	20	49	23
Stakeholder Balancing	M7	41	45	6
	M8	34	51	7

A stacked percentage area chart show in Fig. 5 is generated based on the data from Table III, the chart showed us the constituent parts of a whole have changed for each metrics, and we can have an overall performance and acceptance of the process implementation. From the stacked area chart, we can see that the metric M1, M3, M7, M8 have the highest positive score and lowest negative score while M5 and M6 have low positive score and high negative score. Based on the chart we can say that the convergence time to new process is considered fast and the performance of benefits of improved efficiency during project execution is considered excellent. During the process implementation, resolution of the conflict and debate between the stakeholders and the time to create new relationship after the implementation are considered efficient. However, the time spent and resources invested on the new process construction and implementation is considered high and expensive.

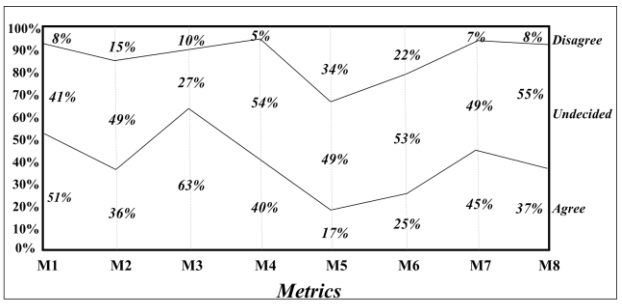

Figure 5. Metrics evaluation.

The responses summary data was then further break down to each question. All respondent and the answered items are then summed and not reversed-scored. For each response, the score of item "agree" will be set to 1, "undecided" to 0 and "disagree" to -1. Each question is numbered based on their corresponding metric, summed and calculated the average score. With the summed and average score of each question, we can further analyze the data for each criterion to determine the performance and impediments and how we can further improve the process.

The responses summary data was then further break down to each question. All respondent and the answered

items are then summed and not reversed-scored. For each response, the score of item "agree" will be set to 1, "undecided" to 0 and "disagree" to -1. Each question is numbered based on their corresponding metric, summed and calculated the average score. The undecided item will not be factored in the calculation. The questionnaire scored chart is summed and averaged as shown in Table IV. Based on the summed scored and calculated data, four charts were generated based on each criterion. With the summed and average score of each question, we can further analyze the data for each criterion to determine the performance and impediments and how we can further improve the process.

TABLE IV. QUESTIONNAIRE SCORED CHART

Criteria	No.	Sum	Avg.	Type
Process Integration	M1-1	14	0.61	positively-keyed
	M1-2	-13	-0.57	negatively-keyed
	M1-3	13	0.57	positively-keyed
	M1-4	0	0.00	negatively-keyed
	M2-1	9	0.39	negatively-keyed
	M2-2	-8	-0.35	negatively-keyed
	M2-3	19	0.83	positively-keyed
	M2-4	-1	-0.04	negatively-keyed
Process Quality	M3-1	15	0.65	positively-keyed
	M3-2	8	0.35	positively-keyed
	M3-3	18	0.78	positively-keyed
	M3-4	8	0.35	positively-keyed
	M4-1	8	0.35	positively-keyed
	M4-2	8	0.35	positively-keyed
	M4-3	5	0.22	positively-keyed
	M4-4	11	0.48	positively-keyed
Process Implementation	M5-1	6	0.26	negatively-keyed
	M5-2	-1	-0.04	negatively-keyed
	M5-3	11	0.48	negatively-keyed
	M5-4	-1	-0.04	negatively-keyed
	M6-1	-4	-0.17	negatively-keyed
	M6-2	10	0.43	negatively-keyed
	M6-3	-5	-0.22	negatively-keyed
	M6-4	-4	-0.17	negatively-keyed
Stakeholder Balancing	M7-1	6	0.26	positively-keyed
	M7-2	9	0.39	positively-keyed
	M7-3	10	0.43	positively-keyed
	M7-4	10	0.43	positively-keyed
	M8-1	8	0.35	positively-keyed
	M8-2	7	0.30	positively-keyed
	M8-3	10	0.43	positively-keyed
	M8-4	2	0.09	positively-keyed

VI. CONCLUSION AND SUGGESTION

The purpose of this study provides an introduction on the implementing of the automation model in continuous integration based on the literature review, deployment and evaluation on the performance and acceptance of the proposed continues integration of agile software development. Based on the data collected from the case study and the analysis performed, conclusions were drawn from the results of this study.

First, the proposed model appears to have the effect on reducing the overall cost of a software project. The data analysis shows that during the software development life cycle, problems can surface and resolve more quickly and there are less problems with deployment, we can conclude that the hidden cost on dealing with unexpected problems might be reduced. Second**,** the proposed model appears to have a positive implication on the increase of productivity, the definition of productivity for development teams in software project are usually the quantity of the code produced under a specific amount of time. The data analysis shows that after the automation process deployed, dealing with change of requirements can be more efficient, the code can be delivered more quickly and less time are used on resolving problems, which can all be contribute to the better productivity. Third**,** communication and collaboration with stakeholders receive very positive responses from the survey, the use of common tools and frequent update of information seems to be the main contributors. The value of better communication and collaboration can be critical factor for improving the effectiveness and efficiency of project.

However, the proposed model does post challenges. Based on the evaluation, during the initial phase the higher learning curve to the implementation and the additional time/resources needed to be invested on the constructing and integration of multiple tools, might cause unforeseen problems, which should be taken into consideration when planning the adoption. Finally, it contributes to the delivery of higher quality software and more efficient process in software development, achieving the software project on schedule and on budget.

REFERENCES

[1] J. Johnson, *CHAOS Report*, the Standish Group, 2015.

[2] T. Dybå, "Improvisation in Small Software Organizations," *IEEE Software*, vol. 17, no. 5, pp. 82-87, 2000.

[3] M. Virmani, "Understanding DevOps & bridging the gap from continuous integration to continuous delivery," in *Proc. Fifth International Conference on Innovative Computing Technology*, Pontevedra, Spain, May 20-22, 2015.

[4] N. Rathod and A. Surve, "Test orchestration a framework for continuous integration and continuous deployment," in *Proc. International Conference on Pervasive Computing*, Pune, India, Jan. 8-10, 2015.

[5] T. Dingsøyr, S. Nerur, V. Balijepally, and N. B. Moe, "A decade of agile methodologies: Towards explaining agile software development," *Journal of Systems and Software*, vol. 85, no. 6, pp. 1213-1221, 2012.

[6] D. Rawsthorne and D. Shimp. (Nov. 2016). Scrum in a nutshell. *Scrum Alliance*. [Online]. available: http://www.scrumalliance.org/articles/151-scrum-in-a-nutshell

[7] J. Winter, K. Rönkkö, M. Ahlberg, and J. Hotchkiss, "Meeting organisational needs and quality assurance through balancing agile and formal usability testing results," in *Proc. the 3rd IFIP TC2 Central and East* European *Conference on Software Engineering Techniques*, 2008, pp. 275-289.

[8] T. Khalane and M. Tanner, "Software quality assurance in scrum-the need for concrete guidance on SQA strategies in meeting user expectations," in *Proc. IEEE Adaptive Science and Technology*, Pretoria, South Africa, Nov. 25-27, 2013.

[9] G. Booch, *Object-Oriented Analysis and Design with applications* 2nd edition, Addison Wesley Longman, 1994.

[10] M. Fowler. (2006). Continuous Integration. [Online]. available: http://martinfowler.com/articles/continuousIntegration.html

[11] S. T. Lai and F. Y. Leu, "Applying continuous integration for reducing web applications development risks," in *Proc. 10th*

International Conference on Broadband and Wireless Computing, Communication and Applications, 2015, pp. 386-391.
[12] L. Chen, "Continuous delivery huge benefits, but challenges too," *IEEE* Software, vol. 32, no. 2, pp. 50-54, 2015.
[13] M. Soni, "End to end automation on cloud with build pipeline: The case for DevOps in insurance industry, continuous integration, continuous testing, and continuous delivery," in *Proc. 2015 IEEE International Conference on Cloud Computing in Emerging Markets*, 2015, pp. 85-89.
[14] Puppet Labs. (Sep. 2014). 2013 state of DevOps report. [Online]. available:http://puppetlabs.com/w-pcontent/uploads/2013/03/2013-state-of-devops-report.pdf
[15] J. Davis and K. Daniels, *Effective DevOps: Building a Culture of* Collaboration, *Affinity, and Tooling at Scale*, O'Reilly Media, Inc., 2016.
[16] B. Mutschler and J. Bumiller, "Towards an evaluation framework for business process integration and management," in *Proc. 2nd International Workshop on Interoperability Research for Networked Enterprises Applications and Software*, Enschede, 2005.

Hsin-Ke LU is the Director of Department of Information Management and the Dean of School of Continuing Education at Chinese Culture University in Taiwan. He is the Chairman of the Association of Continuing Education of Colleges and Universities in Taiwan and the Chairman of Cisco Networking Academy. He is also a holder of Open Group Certification for Enterprise Architecture and Certification in the Enterprise Architect (TOGAF). His academic interests focus on e-learning, corporate structure, lifelong learning and information system planning (Enterprise Architecture).

Peng-Chun LIN is an Assistant Professor of Department of Information Management in Chinese Culture University and the Ph.D. candidate of Information Education Research Institute of National Taiwan Normal University. She is also the Chief Officer of the International Information Certification Cooperation Center at Chinese Culture University in Taiwan. Her academic interests focus on e-learning, network community development, application of simulation learning and recently on optimal network management and corporate structure.

Pin-Chia Huang is the Manager of Application Technology Service Department in Asgard System, Inc., a Taiwan based system Integration Company. He also received his master degree from the Department of Information Management at Chinese Culture University in January 2017.
His academic interests focus on optimizing software development lifecycle and delivery process planning. His professional specialties include open source integration, software architecture design/planning and with proficiency in end-to-end software project management.

An Yuan is the R&D Engineer of Division of Educational Technology Business, School of Continuing Education at Chinese Culture University. He graduated from Chinese Culture University in 2017 and got a master's degree in Information Management. His research field lies systems development and software architecture design/planning.