Continuous Testing Improvement Model

Maximiliano Agustin Mascheroni
Departamento de Informática, FaCENA
Universidad Nacional del Nordeste
Corrientes, Argentina
Universidad Nacional de La Plata
La Plata, Buenos Aires, Argentina
mmascheroni@unne.edu.ar

Emanuel Irrazábal
Departamento de Informática, FaCENA
Universidad Nacional del Nordeste
Corrientes, Argentina
eirrazabal@unne.edu.ar

Gustavo Rossi
LIFIA, Facultad de Informática
Universidad Nacional de La Plata
La Plata, Buenos Aires, Argentina
gustavo@lifia.info.unlp.edu.ar

***Abstract* —Continuous Delivery is a practice where high-quality software is built in a way that it can be released into production at any time. However, systematic literature reviews and surveys performed as part of this Doctoral Research report that both the literature and the industry are still facing problems related to testing using practices like Continuous Delivery or Continuous Deployment. Thus, we propose Continuous Testing Improvement Model (CTIM) as a solution to the testing problems in continuous software development environments. It brings together proposals and approaches from different authors which are presented as good practices grouped by type of tests and divided into four levels. These levels indicate an improvement hierarchy and an evolutionary path in the implementation of Continuous Testing. Also, an application called EvalCTIM was developed to support the appraisal of a testing process using the proposed model. Finally, to validate the model, an action-research methodology was employed through an interpretive theoretical evaluation followed by case studies conducted in real software development projects. After several improvements made as part of the validation outcomes, the results demonstrate that the model can be used as a solution for implementing Continuous Testing gradually at companies using Continuous Deployment or Continuous Delivery and measuring its progress.**

Keywords—continuous testing, continuous delivery, continuous deployment, doctoral thesis

I. INTRODUCTION

Recent studies have shown that there are still testing problems in software development projects that use practices such as Continuous Integration, Continuous Deployment or Continuous Delivery. Some of these problems are [1]: integration tests, automated tests that generate false positives, automated GUI tests that are difficult to maintain, high test execution time, and lack of coverage of certain types of tests in continuous development environments. Furthermore, the industry has reported problems related to the testing process in continuous environments [2]. These are time-consuming tests, unstable tests, flaky tests, automated graphical interface tests, Big Data tests, and a lack of procedures, patterns, and good practices for automated testing in continuous development environments. This shows that both the academic literature and the industry are aligned.

According to Prusak, “the industry has not closed the circle yet when it comes to realizing a full Continuous Delivery Process” [3]. Also, some authors state that Continuous Testing is the missing component in continuous development approaches [4], [5]. As part of Continuous Testing, a large number of techniques, methods and tools have been proposed to deal with the problems related to testing in continuous environments. However, implementing this process has been challenging in practice [1].

For that reason, the Continuous Testing Improvement Model is proposed as a solution to the testing process problems in Continuous Deployment or Continuous Delivery. The model has been created as part of a Doctoral Thesis Research with the collaboration of international companies that develop different types of platforms and systems.

Apart from this introductory section, Section II lists the most critical problems in continuous testing. In section III the proposed model is detailed. The validation of the model is described in section IV. Finally, conclusions are presented in section V.

II. TESTING PROBLEMS

A. *Testing Problems Reported by Academic Sources*

First, as part of the Doctoral Research, a Systematic Literature Review (SLR) has been carried out, searching for testing problems and solutions for them in Continuous Delivery [4].

The selection process resulted in 56 studies that met the inclusion criteria. After the data were extracted and analyzed, a list of 8 problems was obtained for Continuous Testing: cloud service-based application testing, challenges with GUI testing, continuous monitoring, Testing as a Service in Continuous Delivery, automated microservices testing, flaky tests, Big Data testing and automated testing of dynamic websites. Fig. 1 shows the number of problems reported by the studies of the SLR.

B. *Testing Problems Reported by the Industry*

With the aim of searching, identifying, and providing information on the status of testing processes in software development projects that use Continuous Integration and similar practices such as Continuous Deployment or Continuous Delivery, a survey was carried out [2]. Its goal was to validate whether testing challenges or problems exist at the industry and to reveal possible solutions for them.

After a four-month survey period, a total of 287 projects responded. However, following a set of inclusion criteria for handling inconsistent and incomplete questionnaires, 255 of them were kept. After extracting the data and analyzing the results, the testing problems found are listed in TABLE I.

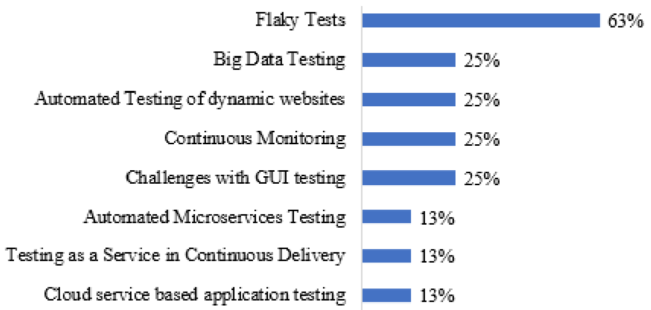


Fig. 1. Testing Problems reported in the academic literature.

TABLE I. TESTING PROBLEMS AT THE INDUSTRY

Testing Problem	Nª of Projects
Automated testing of applications composed by cloud services	2
Mobile testing	17
Big Data testing	18
Non-functional automated testing	23
Data testing	26
Unstable environments	36
Dynamic Web UI automated testing	47
Web service testing	58
Ambiguous test results	65
Lack of procedures, patterns and good practices for automated testing in Continuous Delivery	112
Time-consuming testing	179
Flaky Tests	224
Automated GUI Testing	1234

The previously problems mentioned represent difficulties in implementing the testing process in continuous development approaches correctly. Besides, they are related to each other in such a way that the occurrence of one produces another directly or may produce it. These relationships are shown in Fig. 2.

Finally, it can be seen from Fig. 2 that the most important problems are time-consuming tests, flaky tests and lack of frameworks, tools and good practices for testing in continuous development approaches.

III. CONTINUOUS TESTING IMPROVEMENT MODEL

In the academic literature and in industry examples, many approaches have been proposed to deal, at least partially, with each of the mentioned problem. In this way, by gathering and analyzing all these proposals, it is possible to build a set of well-structured practices to gradually adopt Continuous Testing inside an organization. This set of structured good practices grouped together is proposed as a model with adoption levels called improvement levels. This concept is used in other process improvement models like CMMI [7] or TMMi [8]. The proposal is called **Continuous Testing Improvement Model** (CTIM).

CTIM proposes four levels of improvements and five verification and validation (V&V) stages that encompass the proposals, tools and techniques reported as a set of good practices.

The improvement level 1 of the model is called **implementation**. It proposes test automation as the first step in the adoption of Continuous Testing and the initial implementation of each V&V stage which generates the Continuous Testing Pipeline (Fig. 3).

The improvement level 2 is called **management**. It implements good practices related mainly to the management of tests. Unit and functional tests are grouped. Unit testing coverage is measured, and the functional testing framework is layered for easy maintainability. The use of a farm for mobile device testing and the adoption of exploratory and capacity testing is also proposed. This level largely solves the problems of low-testing coverage in Continuous Delivery.

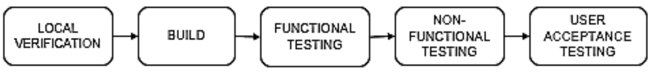
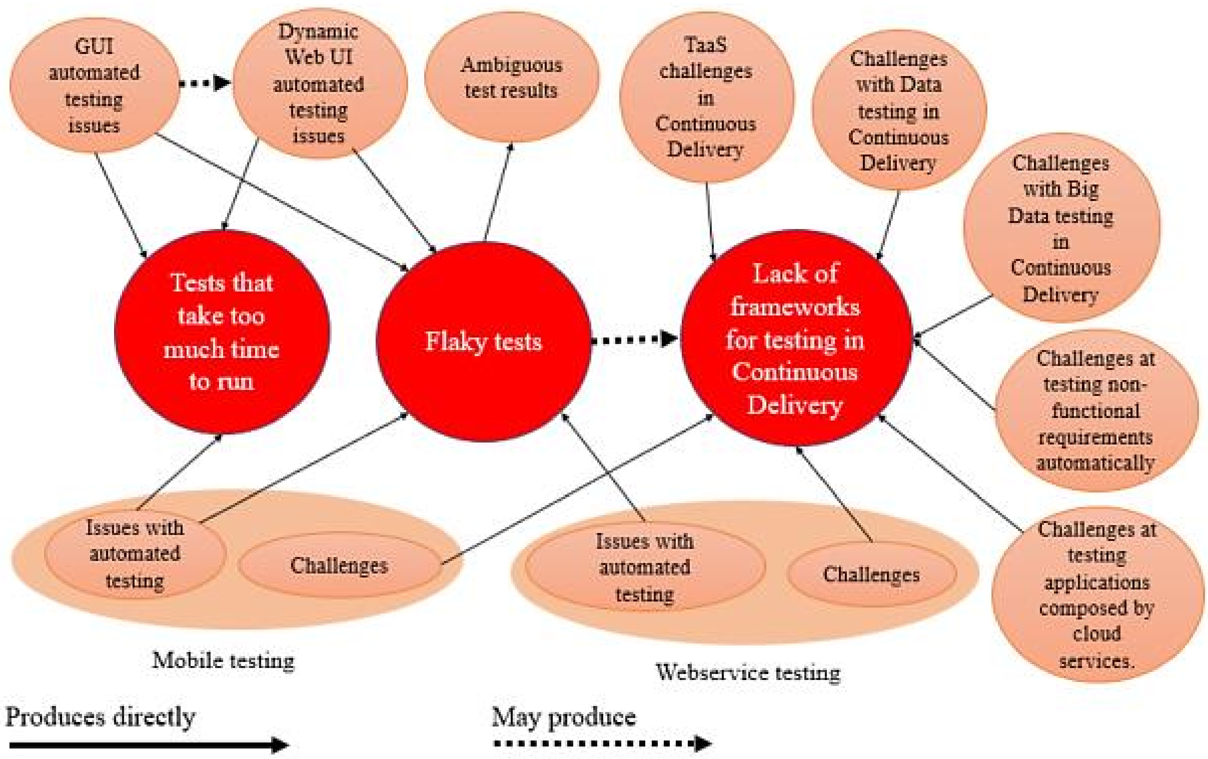


Fig. 3. Continuous Testing Pipeline.



Fig, 2. Relationship between testing problems [6].

The improvement level 3 of the model is called **reliability**. The problem that is mainly solved at this level is flaky tests. Standards and patterns are defined for a testing process that generates reliable results. Good practices include the implementation and execution of test mocks, headless functional tests, static code analysis and deployment tests. This level also encompasses waiting and skipping strategies for the tests and suggestions for improving test reports.

Finally, the improvement level 4 is called **continuity**. Practices such as test selection, parallelization, use of APIs to execute preconditions, browser rotation, among others, are implemented. At this level, the problem of time-consuming testing is solved. It also seeks to incorporate continuous improvement mechanisms for the different types of tests.

A. Validation and Verification Stages

CTIM includes a set of V&V stages that encompass different types of tests that have been defined and proposed by different authors both in the industry and in the academic literature. These V&V stages shape the Continuous Testing Pipeline, analogous to the concept of Deployment Pipeline and which is shown in Fig. 3. The stages are:

- **Local Verification**: It occurs before the code is integrated into the main trunk, from the version control repository.
- **Build**: It is the first stage running on the Continuous Integration server. Its objective is to affirm that the system works on a technical level.
- **Functional Testing**: It covers both GUI and API tests that verify compliance with the functional requirements of the system.
- **Non-Functional Testing**: It covers the different types of tests related to non-functional requirements such as security, performance, etc.
- **User Acceptance Testing**: This stage covers activities that cannot be automated: exploratory testing by testing experts, customer software demonstrations approvals by stakeholders, etc.

TABLE II summarizes the proposed model and lists all the good practices that comprise it.

TABLE II. CONTINUOUS TESTING PRACTICES IN CTIM

CTIL	V&V Stage	Continuous Testing Practice
1	Local Verification	Local Unit Testing
		Code Reviews
	Build	Automated Build
		Automated Unit Tests Execution
	Functional Testing	Automated Functional Testing
		Functional Testing Coverage
	Non-Functional Testing	Performance Testing
	User Acceptance Testing	Cross-Browser Testing
		Showcases and Demonstrations
2	Local Verification	Unit Tests Grouping
		Unit Tests Coverage
	Build	Automated Deployment
	Functional Testing	Multi-Layered Framework
		Functional Test Segmentation
		Mobile Device Farm
	Non-Functional Testing	Capacity Testing Management
	User Acceptance Testing	Exploratory Testing Management
3	Local Verification	Test Doubles
		Headless Functional Testing
	Build	Static Code Analysis
		Installation/Deployment Testing
	Functional Testing	Non-Happy path Testing
		E2E Testing
		Test Waiting Strategy
		Test Skipping Strategy
		Re-running Test Failures
		Scheduled Tests Execution
		Test Data Generation
		Generation of accurate reports
	Non-Functional Testing	Security Testing Management
	User Acceptance Testing	Usability Testing
4	Local Verification	Running Unit Tests in the background
	Build	Parallel Unit Testing
		Measuring build scalability and performance.
	Functional Testing	Parallel Functional Testing
		Functional Tests Selection
		Using APIs for preconditions
		Browser Rotation
		Image Comparison
		Continuous Monitoring
	Non-Functional Testing	Automated Capacity Testing
	User Acceptance Testing	Crowdsourced Testing

Finally, CTIM is shown in Fig. 4.

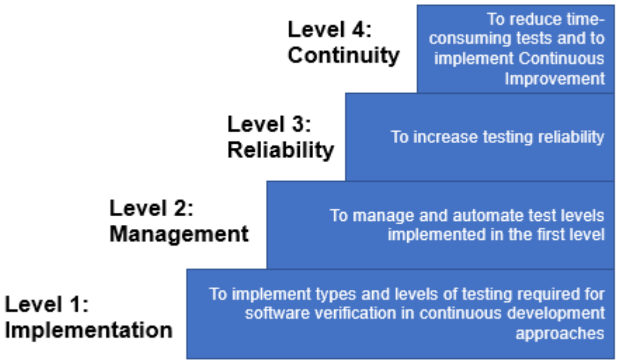


Fig. 4. Continuous Testing Improvement Model.

IV. VALIDATION

The validation of the model has been carried out using the Action-Research method [9]. In this method, a validation process is composed of groups of activities organized in a cycle. The process used for validating the model was divided into two phases (Fig. 5). The input is the proposed model and the output is the validated model, which was presented in Section III.

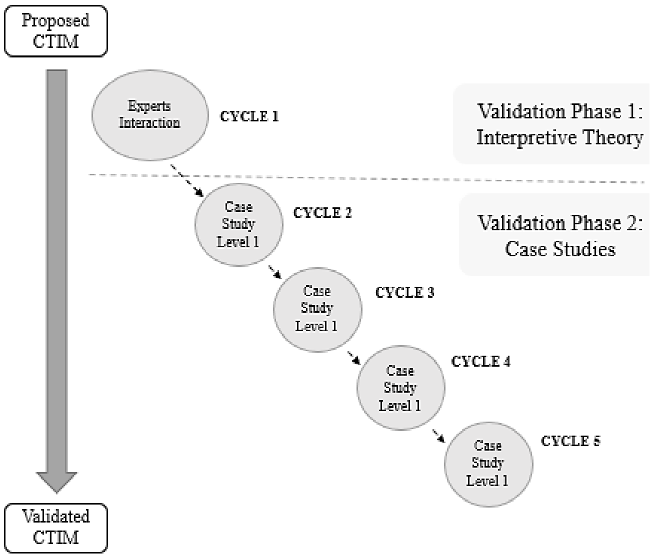


Fig. 5. Action-Research cycles for the validation of CTIM.

In the first theoretical validation, 17 experts reviewed the model and then proposed improvements. Most of the experts have more than 10 years of experience working with Continuous Development practices and the rest have research experience in related fields. After that, the model was implemented in 10 projects belonging to companies that develop software. That implementation generated results which generated 30 improvements. The improvements are summarized in TABLE III.

The implementation of the model was supported with a tool called EvalCTIM[1] which was developed as part of the Doctoral Thesis Research to assess an improvement level using CTIM.

TABLE III. CONTINUOUS TESTING PRACTICES IN CTIM

Level	Improvements	New Practices	Modified Practices	Deleted Practices
1	6	0	4	0
2	6	1	3	0
3	11	4	5	0
4	7	1	4	0

V. CONCLUSION

In this paper, the Continuous Testing Improvement Model was presented as part of a Doctoral Thesis Research. The model was proposed as a solution for the reported testing problems in projects which are using continuous development approaches.

The model was validated using an Action-Research method divided into two phases. Both the experts who were selected for the first validation (phase 1) and the specialists who participated in the implementation of the model (phase 2) agreed that the model is a solution to the Continuous Testing problems that exist today.

REFERENCES

[1] E. Laukkanen, J. Itkonen, and C. Lassenius, “Problems, causes and solutions when adopting continuous delivery — A systematic literature review,” Information and Software Technology, vol. 82, pp. 55–79, Feb. 2017, doi: 10.1016/j.infsof.2016.10.001.

[2] Author X. Paper X. Congreso Argentino de Ciencias de la Computación (CACIC), 2019.

[3] O. Prusak, “Continuous testing: The missing link in the continuous delivery process.” https://www.blazemeter.com/blog/continuous-testing-missing-link-continuous-delivery-process (accessed Mar. 04, 2021).

[4] Author X. Paper X. Computación y Sistemas, vol. 22, no. 3, Sep. 2018.

[5] BlazeMeter, “Continuous Testing in practice. Completing the Continuous Delivery Process.” 2015, [Online]. Available: https://info.blazemeter.com/shift-left-testing.

[6] Author X. Paper X. Congreso Argentino de Ciencias de la Computación (La Plata, 2018).

[7] “CMMI Institute - CMMI Development.” https://cmmiinstitute.com/cmmi/dev (accessed Aug. 24, 2020).

[8] “TMMi Model – TMMI.” https://www.tmmi.org/tmmi-model/ (accessed Aug. 25, 2019).

[9] A. T. Wood-Harper, “Research Methods in Information Systems: Using Action Research,” in Research Methods in Information Systems, pp. 169–191.

[1] http://ctim.com.ar/evalctim