ISSN 2277-3061
Volume 10 Number 11
International Journal of Computers & Technology

Choosing Automated or Manual Testing in Extreme Programming with the Analytical Hierarchy Process

Sultan Alshehri, Abdulmajeed Aljuhani and Luigi Benedicenti
Department of Engineering, Software Engineering, University of Regina, Canada
aljumais@uregina.ca
aljuhana@uregina.ca
luigi.benedicenti@uregina.ca

ABSTRACT

Extreme Programming (XP) has been called one of the most successful methods in software development. XP comprises a set of practices designed to work together to provide value to the customer. During the XP lifecycle, developers and customers regularly encounter situations in which they need to make decisions or evaluate factors. This affects the development process and team productivity. We propose to use the Analytic Hierarchy Process (AHP) as a means to systematize and streamline the decision process. AHP eliminates conflict because it elaborates input from every member of the team. Thus, the adoption of AHP can help accomplish XP values and fulfill team needs. This paper presents an example of applying the AHP to decide which testing technique to adopt depending on a series of project-specific parameters.

KEYWORDS

Extreme programming, Test-Driven Development; Testing techniques; Analytic Hierarchy Process.

INTRODUCTION

Test-Driven Development (TDD) is an XP practice where unit test cases are incrementally written before implementation takes place. It was first described in detail by Beck and successfully adapted to numerous agile projects [1]. TDD is a valuable part of XP as it is associated with a range of benefits such as increasing programming productivity and speed, producing high design quality, and reducing the work required for fixing defects [2].

Unit tests in XP projects are often automated and must pass every time. Tests are written just before starting the code. Test automation in XP helps save time and effort, and reduce the long-term cost of testing. Automated tests reduce the regression test cycle and can be performed in parallel on multiple machines. In addition, the testing tool can be used for both unit and acceptance testing [3].

Automated testing is at the heart of TDD, but it can be more expensive than manual testing [4]. Thus, an XP team should not perform automated tests for the sake of automation alone. There are situations when one needs to conduct semi-automated tests or even manual ones. For example, an agile tester from VersionOne, believes that if a test is only performed once it should not be automated: “One-time tasks and exploratory testing/edge cases shouldn’t be automated. Edge cases are, by definition, usually one-off test cases, and the effort to automate edge cases generally does not pay off. Exploratory tests are best used to gain knowledge of a new feature and then tweak or revise tests based on the new knowledge” [5].

Some of the risks associated with automated tests are as follows: (1) the development team may rely too much on automation as the primary indicator of quality; (2) the team might disconnect from the application and lose sight of the user; (3) applications can become less intuitive [6]. There are also specific cases where testing should not be handled through automation. For example, ad-hoc/exploratory testing, session based testing, vulnerability/security testing, and negative testing should not be automated because changing the data manually to reflect negative behavior and edge cases are best handled manually due to the complexity that is often involved [6].

Lisa Crispin criticized manual testing in her book, saying, “When manual testing is subjected to schedule pressure, the quality of the testing goes down. People begin to cut corners, omit tests, and miss problems. This is the kind of dysfunctional behavior for which traditional software development is famous. All those manual test cases look great on paper, but when crunch time hits, all the paper goes out the window” [7].

On the other hand, Naidu said, “While automation continues to evolve rapidly, it is too early in the technological revolution to replace manual testing completely with automation. In fact, most of the new features, complex validations and business intensive functionalities will continue to be tested manually. The goal of 100% automation is not just ambitious, it is also impractical. Typically 70% automation helps maximize return on investments. Hence, manual testing remains and will continue to dominate in organizations with lower levels of automation maturity and in areas where ROI on automation is not significant [sic]” [8].

He also continued to highlight some important issues about automated testing:

“Automation is not just a one-time initiative. Failure rates for automation are higher than success rates, and therefore careful planning is essential. Automation skills and tools are expensive” [8].

Johnson agreed, and wrote about the importance of skills in automated testing: “so, does automation replace the need for testers? No. Does automation change the role and skills required to fulfill the quality verification task? Yes” [9].

Manual testing can be very time consuming, but testers are free to explore and attempt to break the system, while that is not possible in automated testing as its goal is not to break the system but to notify when a change in the code broke a test.

As a result, having automated testing in an organization can be a somewhat controversial decision. Many automation efforts fail or do not meet the expected return on investment [9].

THE ANALYTICAL HIERARCHY PROCESS

AHP is a multi-criteria decision making tool that structures a problem in a hierarchical model. Human factor plays a main role in AHP by gathering the elements of a problem and structuring them hieratically. Thomas Saaty has introduced AHP as a robust and sufficient method that can be used to solve complex decision-making problems [10]. AHP consists of several steps, as follows:

- Breaking down the problem into hierarchal model.
- Identifying the criteria and designing criteria pairwise comparison matrix. Thus, the weight for each criterion will be specified with respect to the other criteria.
- Designing alternative pairwise matrices with respect to the control criterion in each matrix.
- Calculating the consistency ratio.
- Selecting the highest weighted average rating after calculating the average weight for each alternative.

In order to facilitate pairwise comparisons, Saaty [11] presented an importance scale for each criterion and alternative. Table 1 shows this importance scale.

Table 1 AHP Numerical Scale Developed by Saaty [11].

Scale	Numerical Rating	Reciprocal
Equal importance	1	1
Moderate importance of one over other	3	1/3
Very strong or demonstrated	7	1/7
Extreme importance	9	1/9
Intermediate values	2,4,6,8	1/2, 1/4, 1/6, 1/8

CASE STUDY SETUP

A part of this work has been published in [12]; therefore, the case study setup described below is similar to that in [12].

Research Questions and Propositions:

The primary objective in the test-driven development practice is to investigate how AHP can be used in selecting the best testing techniques and ranking the release indicators. The following research questions provided a focus for our case study investigation:

- How important is it to practice the test-driven development using AHP?
- How can AHP rank the testing techniques based on specific criteria?
- How can AHP help reducing the cost of executing tests?
- How can AHP help in saving the developers' time while testing?

The methodology used in this study is the case study methodology. We conducted four case studies: two case studies in an academic environment and two case studies in industrial environments with embedded units of analysis. The study propositions are outlined below:

- AHP captures important criteria and alternatives that need to be considered when testing.
- AHP facilitates the process of ranking and selection in testing.
- AHP involves an informative discussion and improves communication among the developers.
- AHP provides a map to focus on the most important testing techniques that reduce the testing execution cost.

Unit of Analysis:

According to [13] the unit of the analysis should be derived from the main research questions of the study. The main focus of this study is to select the best testing techniques based on specific criteria. So, the selection of a testing technique and the evaluation process are the first two units of analysis for this study. The third is the developers' view of how AHP benefits each XP practice. As result, we designed the four case studies as multiple cases (embedded) with multiple units of analysis.

Data Collection and Sources:

Literature review and previous studies provided the starting point for our investigation. To increase the validity of this study, data triangulation was obtained. The data sources in this study were:

- Archival records, such as a study plan from the graduate students.
- Questionnaires given to the participants when developing an XP project.
- Questionnaires given to experts from industry.
- Open-ended interviews with the participants.
- Feedback from the customer.

However, the most important data source of this study was an XP project conducted at the University of Regina in a software design class in the Fall of 2012. To complement the study, three companies participated in evaluating some of the XP practices. Afterwards, two of the three companies were involved in validating the results.

Designing the Case Studies:

The educational case studies were performed as part of a course in the Advanced Software Design Class for graduate students taught in Fall 2012 at the University of Regina. The participants were 12 master's students and a client from a local company in Regina. Participants had various levels of programming experience and a good familiarity with XP and its practices. The students' background covered several programming languages such as Java, C, C#, and ASP.net.

All the students had previous experience implementing projects using various software process methodologies. The study was carried out throughout 15 weeks; and the students were divided into two teams. Both teams were asked to build a project called "Issue Tracking System" brought by the client along with industrial requirements. The project ran in 5 main iterations and by the end of the semester, the whole software requirements were delivered. The students were paired based on their experience and knowledge, but also we had an opportunity to pair some experts with novice and average programmers for the purpose of the study. Participants were given detailed lectures and supporting study materials on extreme programming practices that focused on planning game activities which included writing user stories, prioritizing the stories, estimating process parameters, and demonstrating developers' commitments. The students were not new to the concept of XP, but they gained more knowledge and foundations specifically in the iteration plan, release planning and prioritizing the user stories. In addition, the students were exposed to the AHP methodology and learned the processes necessary to conduct the pairwise comparisons and to do the calculations. Several papers and different materials about the AHP and user stories were given to the students to train them and increase their skills in implementing the methodology. In addition, a survey was distributed among students to get further information about their personal experiences and knowledge.

The researchers visited three companies (two companies in Regina, and one in Calgary; both in Canada) several times and met with the developers and team leaders to explain the purpose of the study and to collect the data and feedback from the real industries. To preserve their anonymity, the letters A, B, and C replace the companies' names in this work. All the companies were familiar with XP concepts at the time of the study and are currently practicing Test-Driven Development during their development. In this study, eighteen experts have used their knowledge and average of 10 years experience to evaluate the proposed pair alternatives using the AHP.

USES OF AHP IN TESTING

From our review on automated testing, briefly summarized above, we determined that there are several different factors that can play significant roles in the choice of the level of automation in testing. These factors are: 1) time constraint, 2) developers knowledge or understanding, 3) cost of test execution, 4) testing coverage, 5) resource availability, and 6) dealing with GUI applications. Given these factors, AHP can be applied to help select one of the following testing techniques: automated testing, semi-automated testing, and manual testing.

Automated Testing Decision

AHP is used to decide the level of testing based on the criteria elaborated from the factors above. The essential criteria that affect the decision are then as follows.

Proposed criteria for the automation decision

- *Time:* which testing option is best in terms of spending less time in the testing phase?
- *Developer Understanding:* which testing option is easier for the developers to understand and apply?
- *Cost of execution:* which testing option costs less when executed?
- *Test coverage:* which testing option can cover more of the source code that has been tested?
- *Resource:* which testing option is best in terms of resource consumption?

- *GUI:* which testing option is best when dealing with GUI testing?

AHP-TEST DECISION STRUCTURE

The AHP-Test Decision Structure includes three levels. The top level is the main objective, which in our case is deciding the level of testing; the second level is the criteria, which in our case are as follows: time, developer understanding, cost of execution, test coverage, resources, and graphic user interface; and the third level is the decision options (i.e., outcomes), which in our case are as follows: automated testing, semi- automated testing, and manual testing. Figure 1 illustrates the AHP structure for the problem.

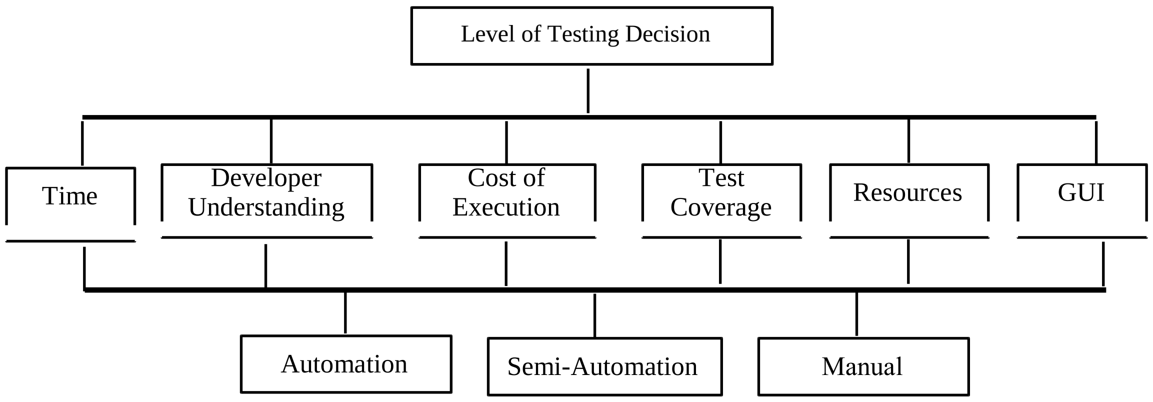

Figure 1 AHP Structure for testing decision.

Pairwise Comparison Process for the Automation Decision

In accordance with the AHP, all participants were asked to evaluate the outcomes based on the criteria. For this purpose, sheets of paper with appropriate AHP tables were handed to the all students in order to facilitate the comparison process. The participants first compared the criteria using the Saaty scale. The participants were asked the following questions:

- Which is more important: time or developer understanding and by how much?
- Which is more important: time or cost of execution and by how much?
- Which is more important: time or test coverage and by how much?
- Which is more important: time or resource and by how much?
- Which is more important: time rogue testing and by how much?
- Which is more important: developer understanding or cost of execution and by how much?
- Which is more important: developer understanding or test coverage and by how much?
- Which is more important: time or developer understanding and by how much?
- Which is more important: time or cost of execution and by how much?
- Which is more important: time or test coverage and by how much?
- Which is more important: time or resource and by how much?
- Which is more important: time rogue testing and by how much?
- Which is more important: developer understanding or cost of execution and by how much?
- Which is more important: developer understanding or test coverage and by howmuch?

After finishing the criteria comparisons, the participants proceeded to evaluate all testing techniques based on each criterion. An example follows:

- In term of the time, which testing techniques will save us more time and by how much?

All terms of following comparisons were conducted based on each criterion: (Automation X semi-automation), (automation X manual), (semi-automation X manual).

These questions and comparisons were related until the testing options were rated based on each criterion. The results are shown in the following section.

AHP EVALUATION RESULTS FOR THE AUTOMATION DECISION

1. Educational case studies

For Team 1, the rankings of the testing alternatives based on all criteria, i.e. time, developer understanding, cost of execution, test coverage, resources and GUI, are summarized as follows. First: Automation (38.59); Second: semi-automated (32.00); Third: manually (29.41). Table 2 summarizes the results.

Figure 2 shows the importance of each criterion as follows: code coverage (28.83), cost of execution (19.98), developer understanding (15.01), time (14.75), GUI (11.18) and resources (10.25).

Table 3: Level of testing by team

Testing Techniques	All
Automation	38.95 %
Semi-Automation	32.00 %
Manual	29.41 %

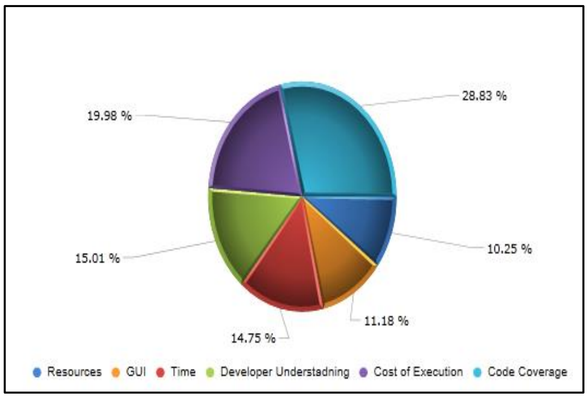

Figure 2 Importance of the criteria for team1

The rankings of the testing alternatives by Team 2 are summarized as follows: First: Automation (44.92); Second: semi-automation (32.00); Third: manual (23.08). Table 2 summarizes the results.

Figure 3 shows the importance of each criterion as follows: cost of execution (34.68), time (20.53), code coverage (20.17), resources (11.19), developer understanding (7.04) and GUI (5.68).

Table 3: Level of testing by team 2

Testing Techniques Automation	All
	44.92 %
Semi-Automation	32.00 %
Manual	23.08 %

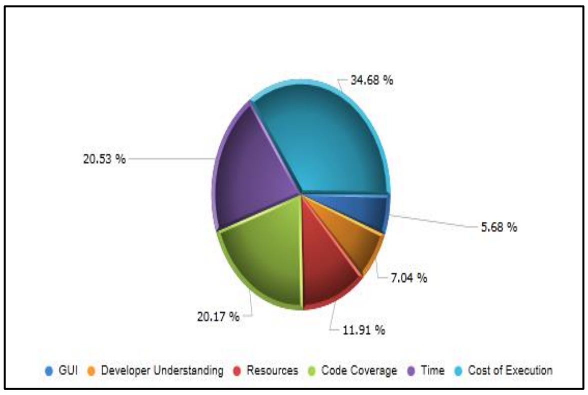

Figure 3 Importance of the criteria for team2

Observations (educational case studies)

1. Considering the criteria as a whole, both teams selected automated testing to be the best option.
2. Team 1 considered code coverage to be the most important criteria, while Team 2 considered the cost of execution to be the most important criteria.
3. Manual testing was ranked as the last option for both teams.
4. If we consider each criterion individually, we can see that Team 1 ranked manual testing as the top option in three criteria: developer understanding, availability of resources, and GUI. Automated testing was ranked as the top option in the other three criteria, time, cost of execution, and code coverage. This is shown in table 4.
5. Team 2 ranked automated testing the highest in terms of time, cost of execution, availability of resources, and GUI. This is shown in table 5.
6. Team 2 ranked manual testing the highest in terms of developer understanding, and ranked semi-automated testing the highest in terms of GUI. This is shown table 5.

Table 4 Testing level by Team 1 based on each criterion individually

Technique	Time	Technique	Developer Understanding	Technique	Cost of Execution	Technique	Code Coverage
Automation	56.52 %	Manual	60.81 %	Automation	47.78 %	Automation	47.21 %
Semi-Automation	23.68 %	Semi-Automation	24.89 %	Semi-Automation	33.11 %	Semi-Automation	30.34 %
Manual	10.80 %	Automation	14.29 %	Manual	19.11 %	Manual	22.46 %

Technique	Resources	Technique	GUI
Manual	51.41 %	Manual	58.02 %
Semi-Automation	33.81 %	Semi-Automation	26.34 %
Automation	14.78 %	Automation	15.64 %

Table 5 Testing level by Team 2 based on each criterion individually

Technique	Time	Technique	Developer Understanding	Technique	Cost of Execution	Technique	Code Coverage
Automation	62.00 %	Manual	69.71 %	Automation	45.51 %	Semi-Automation	39.52 %
Semi-Automation	28.27 %	Semi-Automation	20.24 %	Semi-Automation	32.00 %	Automation	31.36 %
Manual	*9.73 %*	Automation	10.05 %	Manual	22.50 %	Manual	29.12 %

Technique	Resources	Technique	GUI
Automation	65.14 %	Automation	50.75 %
Semi-Automation	26.06 %	Semi-Automation	29.04 %
Manual	8.80 %	Manual	20.21 %

2. Industrial case studies

The rankings of the testing alternatives by company A are summarized as follows: First: Automation (61.27); Second: semi-automation (24.94); Third: manual (13.79).

Table 6 summarizes the results.

Figure 4 shows the importance of each criterion as follows: developer understanding (27.12), cost of execution (21.6), code coverage (16), resources (14.87), time (12.36) and GUI (8.06).

Table 6 Level of testing by team A

Testing Techniques	All
Automation	61.27 %
Semi-Automation	24.94 %
Manual	13.79 %

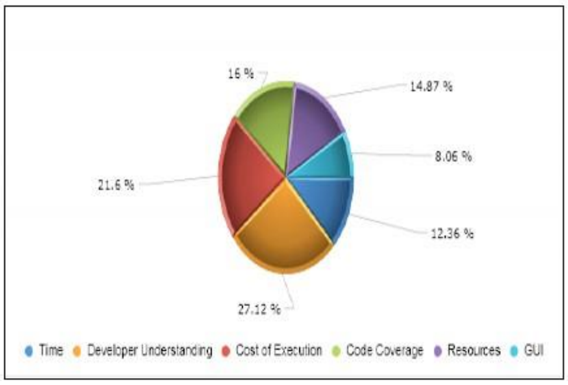

Figure 4 Importance of the criteria for company A

The rankings for the testing alternatives by Company B are summarized as follows: First: Automation (45.09); Second: semi-automation (34.09); Third: manual (20.82).

Table 7 summarizes the results.

Figure 5 shows the importance of each criterion as follows: code coverage(27.85), cost of execution (17.6), resources (16.33), developer understanding (15.16), time (11.55) and GUI(11.51).

Table 7 Level of testing by team B

Testing Techniques Automatio n	All
	45.09 %
Semi-Automation	34.09 %
Manual	20.82 %

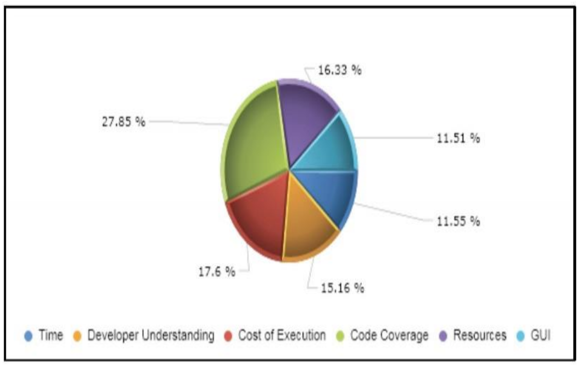

Figure 5 Importance of the criteria for company B

Observations (industrial case studies)

1. Considering the criteria as a whole, the three companies provided the same rankings: the highest rank was automated testing, the second semi-automated testing, and the last was manual testing.
2. Developer understanding was considered the most important criteria for company A and C, while company B considered code coverage to be the highest concern.
3. All three companies ranked the GUI criteria as the least important, showing that it has no significant impact on the decision.
4. If we consider each criterion individually, we can see that all three companies ranked automated testing in the top position for all criteria, with the exception of company B which ranked semi-automation the highest for the GUI criteria. See tables 9, 10, and 11.
5. It is very clear that manual testing was the least preferable option for the three companies.

Table 9 Testing level by company (A) based on each criterion individually

Technique	Time	Technique	Developer Understanding	Technique	Cost of Execution	Technique	Code Coverage
Automation	68.91 %	Automation	63.19 %	Automation	62.21 %	Automation	57.22 %
Semi-Automation	22.09 %	Semi-Automation	22.68 %	Semi-Automation	23.85 %	Semi-Automation	26.71 %
Manual	9 %	Manual	14.12 %	Manual	13.95 %	Manual	16.07 %

Technique	Resources	Technique	GUI
Automation	63.03 %	Automation	50.08 %
Semi-Automation	24.16 %	Semi-Automation	34.52 %
Manual	12.81 %	Manual	15.4 %

Table 10 Testing level by company (B) based on each criterion individually

Technique	Time	Technique	Developer Understanding	Technique	Cost of Execution	Technique	Code Coverage
Automation	64.14 %	Automation	44.87 %	Automation	36.66 %	Automation	50.88 %
Semi-Automation	29.19 %	Semi-Automation	33.98 %	Semi-Automation	35.81 %	Semi-Automation	31.94 %
Manual	*6.67 %*	Manual	21.16 %	Manual	27.53 %	Manual	17.18 %

Technique	Resources	Technique	GUI
Automation	50.79 %	Semi-Automation	40.71 %
Semi-Automation	32.20 %	Automation	29.84 %
Manual	17.00 %	Manual	29.45 %

Table 11 Testing level by company (C) based on each criterion individually

Technique	Time	Technique	Developer Understanding	Technique	Cost of Execution	Technique	Code Coverage
Automation	69.64 %	Automation	51.74 %	Automation	45.59 %	Automation	62.85 %
Semi-Automation	23.42 %	Semi-Automation	29.49 %	Semi-Automation	36.42 %	Semi-Automation	25.67 %
Manual	6.94 %	Manual	18.78 %	Manual	17.99 %	Manual	11.75 %

Technique	Resources	Technique	GUI
Automation	56.86 %	Automation	72.46 %
Semi-Automation	30.86 %	Semi-Automation	21.07 %
Manual	12.28 %	Manual	6.47 %

SEMI-STRUCTURED INTERVIEWS RESULTS

Semi-structured interviews were conducted with students after showing the participants the results of the AHP evaluation. Students found that some of these results were surprising and others were expected. The interviews included open questions to obtain the students' general opinions about the AHP, advantages and disadvantages of the using the AHP, and their opinion on which one of the XP practices would provide the best experience for the application of the AHP. The data was collected in the form of handwritten notes during the interviews. These notes were organized in a folder for the sake of easy access and analysis. From the interviews, we found very positive feedback from the participants regarding the AHP. The AHP resolved any conflicting opinions and brought each team member's voice to the decision in a practical way. It also emphasized the courage of the team by letting every opinion be heard. The time and the number of the comparisons were the main concerns of the participants. All of them recommended using the AHP in the future when it needs to decided which testing technique should be applied. There were a few additional recommendations as well, such as developing an automated tool to reduce the time for the AHP calculation, adding mobility features, performing a cost and risk analysis, and trying the AHP in other XP areas and studying the outcomes.

QUESTIONNAIRES

Questionnaires were given to the participants (students and companies A, B and C) in order to obtain their perceptions of and experiences with the AHP. The questionnaires were divided into two main parts. The first part contained questions about the AHP as a decision and ranking tool. The second part contained questions regarding the direct benefits of the XP practice and investigated the participants' satisfaction. We used a seven-point Likert scale to reflect the level of acceptability of the AHP tool. The seven-point scale was as follows: (1) Totally unacceptable. (2) Unacceptable. (3) Slightly unacceptable. (4) Neutral. (5) Slightly acceptable. (6) Acceptable. (7) Perfectly Acceptable. Once the participants completed the questionnaire, we calculated the results and presented the total percentage of the acceptability for each statement in the evaluation.

The total percentage of the acceptability was calculated as follows:

- o The total percentage of acceptability (TPA)

= The average of the score for each team * 100 / 7.

- o The average of the score for each team

= The sum of the scores given by the team members / number of the team.

Acceptability Level for the Impact of AHP on the Development:

- The following percentages show the acceptability level for the impact of the AHP on the development:

- First: improving team communication; Team 1 (74%), Team 2 (93%), company A (93%), company B (93%), and company C (94%).
- Second: creating an informative discussion and learning opportunities, Team 1 (71%), Team 2 (88%), company A (86%), company B (95%), and company C (93%).
- Third: clarifying the ranking problem; Team 1 (71%), Team 2 (90%), company A (86%), company B (93%), and company C (83%).
- Fourth: resolving the conflicting opinions among members; Team 1 (64%), Team 2 (86%), company A (86%), company B (93%), and company C (83%).
- Fifth: elevating the team performance; Team 1 (76%) and Team 2 (88%), company A (79%), company B (86%), and company C (not study).

VALIDITY

Construct validity, internal validity, external validity and reliability describe common threats to the validity to any performed study [14].”Empirical studies in general and case studies in particular are prone to biases and validity threats that make it difficult to control the quality of the study to generalize its results” [15]. In this section, relevant validity threats are described. A number of possible threats to validity can be identified for this work.

Construct Validity

Construct validity deals with the correct operational measures for the concept being studied and researched. The major construct validity threat to this study is the small number of participants in each case study. This threat was mitigated by using several techniques in order to ensure the validity of the findings, as outlined below.

- Data triangulation: A major strength of case studies is the possibility of using many different sources of evidence [16]. This issue was taken into account through the use of surveys and interviews with different types of participants from different environments with various levels of skills and experience, and through the use of several observations as well as feedback from those involved in the study. By establishing a chain of evidence, we were able to reach a valid conclusion.
- Methodological triangulation: The research methods employed were a combination of a project conducted to serve this purpose, interviews, surveys, AHP result comparisons, and researchers’ notes and observations.
- Member checking: Presenting the results to the people involved in the study is always recommended, especially for qualitative research. This was done by showing the final results to all participants to ensure the accuracy of what was stated and to guard against researcher bias.

Internal Validity

Internal validity is only a concern for an explanatory case study [16]. Internal validity focuses on establishing a causal relationship between students and educational constraints. This issue can be addressed by relating the research questions to the propositions and other data sources providing information regarding the questions.

External validity

External validity is related to the domain of the study and the possibilities of generalizing the results. To provide external validity to this study, we will need to conduct an additional case study in the industry involving experts and developers, and then observe the similarities and the differences in the findings of both studies. Thus, future work will contribute to accruing external validity.

Reliability

Reliability deals with the data collection procedure and results. Other researchers should arrive at the same case study findings and conclusions if they follow the same procedure. We addressed this by making the research questions, case study set up, data collection and analysis procedure plan is available for use by other researchers.

CONCLUSION

After using the AHP to rank the level of testing in an XP project, we was found it to be an important tool that provides a very good vision for developers when they apply it to the testing practice to improve the development process. Considering the time, developer understanding cost of execution, test coverage, resources, and GUI when ranking the level of testing could bring many advantages to the development team such as reducing time and effort and transferring knowledge to the developers. More importantly, though, the AHP had a positive impact on the development process and communication among the team members. The team could mathematically reconcile any conflict of opinions regarding the choice of a level of testing during the whole development cycle. Thus, the AHP provides a cooperative decision making environment, which could maximize the effectiveness of the developed software.

References

[1] Beck, K, “Test-Driven Development by Example”, Addison-Wesley Professional; 1 edition, Nov 2002.

[2] Boby Georgea, Laurie Williams, “A Structured Experiment of Test-Driven Development”,Information and Software Technology, vol. 46, no. 5, April 2004, pp.337–342

[3] Thirumaslesh Bahat , Nachiappan Nagappan, “Evaluating the efficacy of Test-Driven Development: Industrial Case Studies”, in Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering, 2006, pp. 356 – 363.

[4] Brain Marick,”When Should a Test Be Automated”, Reliable Software Technologies, a conference produced by Software Quality Engineering, STAREAST ‘99.

[5] Automated Testing in Agile Environments, <http://www.versionone.com> (accessed 3.5.2013).

[6] What is the role of Manual Testing in an Agile World, Emergence Tech Training, <http://emergencetechtraining.com> [accessed 5.3 2012].

[7] Lisa Crispin, Tip House, “Testing Extreme Programming”, Addison-Wesley Professional Pub, October 25, 2002. [8] Vasudeva Naidu, “Manual Testing vs. Automated Testing: A Decision Point”, Search Software Quality,

<www.searchsoftwarequality.com> (accessed 3.6.2013).

[9] David W. Johnson, “ Is Automated Testing Replacing the Software Tester?” Search Software Quality, 2011. [10] Saaty TL.The Analytic Hierarchy Process, McGraw-Hill, New York, (1980).

[11] Saaty, T.L. How to Make a Decision: the Analytic Hierarchy Process, Interfaces, Vol. 24, No. 6, pp.19--43 (1994).

[12] Alshehri, Sultan. Aljuhani, Abdulmajeed. “Ranking The Refactoring Techniques Based on The External Quality Attributes”. International Journal of Research in Engineering and Science (IJRES). Volume 3 Issue 6 ! June 2015 ! PP.74-87.

[13] Raed Shatnawi, Wei Li, “ An Empirical Assessment of Refactoring Impact on Software Quality Using Hierarchical Quality Model”, International Journal of Software Engineering and Its Applications, vol. 5, no.4, October 2011.

[14] Robert K. Yin, “Qualitative Research from Start to Finish”, The Guilford Press; 1 edition (October 7, 2010)

[15] Rudiger, Lincke, “How do PhD Students Plan and Follow-up their Work? – A Case Study”, School of Mathematics and Systems Engineering, University of Sweden.

[16] Robert K. Yin, “Case Study Research: Design and Methods: Applied Social Research Methods”, SAGE Publications, Inc; 2nd edition (March 18, 1994).