Analysis of Agile and Traditional Approach for Software Development

Harish Rohil
Assistant Professor
Department of Computer Science and Applications
CDLU, Sirsa

Manisha Syan
Research Scholar
Mtech, Computer Science
CDLU Sirsa

Abstract - It has been seen that agile methodology are gaining ground worldwide. Earlier only few researches implemented the use of agile methodologies but nowadays agile processes have become more reliable. Therefore we decided to conduct our own survey. The research objective was to determine the rate of agile approaches usage and the rate of traditional approach usage and practical experience with these approaches in companies. The research was conducted in a Company environment. This paper presents the results of that research.
Keywords – Agile Software Development, Agile Alliance.

I. INTRODUCTION

Agility is dynamic, context specific, aggressively chance-embracing, and growth-oriented process. Agile processes allow responding quickly to requirement changes and stress collaboration between software developers and customers (customer-driven) and early product delivery. Over recent decades, while market forces, systems requirements, implementation technology, and project staff were changing at a steadily increasing rate, a different development style showed its advantages over the traditional one. This *agile* style of development directly addresses the problems of rapid change. A dominant idea in agile development is that the team can be more effective in responding to change if it can reduce the cost of moving information between people, and reduce the elapsed time between making a decision to seeing the consequences of that decision. To reduce the cost of moving information between people, the agile team works to place people physically closer,replace documents with talking in person and at whiteboards, andimprove the team's amicability—its sense of community and morale—so that people are more inclined to relay valuable information quickly. To reduce the time from decision to feedback, the agile team makes user experts available to the team or, even better, part of the team.

A. Agile Software Development

Agile Software Development is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self organizing, cross-functional teams. It promotes the
adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen interactions throughout the development cycle.Software creation can be a complicated process if not done correctly. The key to a successful software project is communication, flexibility and good analysis. That is why agile approach is adopted for software development. It means the flexibility to adapt when things change. It means constant involvement for making solution and then breaking it down into much smaller processes. This means the user is constantly in the loop of software as it progresses. The most important principle in this model is customer satisfaction by giving rapid and continuous delivery of small and useful software. The delivery of the software happens at regular intervals as opposed to after a number of months, which is the case with the waterfall model. The measure of progress used in agile modeling is the different working models delivered to the client. Since the software is developed in small

batches, changes can easily be introduced into the software product. There is a lot of scope for cooperation between the business people and the developers, as the requirements keep coming from the business people at regular intervals. There is a lot of emphasis laid on technical excellence and good design of the software. The software development team has to adapt regularly to the changing circumstances.Agile modeling is a methodology, which makes use of practice for modeling and documentation of software based systems. Traditional modeling methods in software development projects have given way to these practices which are applied in a more flexible manner. It supplements the different agile methodologies like extreme programming, agile unified process and scrum models.

Agile modeling is a methodology, which makes use of practice for modeling and documentation of software based systems. Traditional modeling methods in software development projects have given way to these practices which are applied in a more flexible manner. It supplements the different agile methodologies like extreme programming, agile unified process and scrum models.

B. Traditional Software Development

Software methodologies like Waterfall method, V-Model and RUP are called traditional software development methodologies and these are classified into the heavyweight methodologies. These methodologies are based on a sequential series of steps like requirements definition, solution building, testing and deployment. Traditional software development methodologies require defining and documenting a stable set of requirements at the beginning of a project. There are four phases which are characteristic of traditional software development method. The first step is to set up the requirements for the project and determine the length of time it will take to implement the various phases of development while trying to predict any problems that may arise in the project. Once the requirements are laid out, the next step moves into the design and architectural planning phase where a technical infrastructure is produced in the form of diagrams or models. These bring to the surface potential issues that the project may face as it progresses and provide a workable road map for the developers to implement.

Once the team is satisfied with the architectural and design plan, the project moves into the development phase where code is produced until the specific goals are reached. Development is often broken down into smaller tasks that are distributed among various teams based on skill. The testing phase often overlaps with the development phase to ensure issues are addressed early on. Once the project nears completion and the developers are close to meeting the project requirements, the customer will become part of the testing and feedback cycle and the project was delivered after the customer satisfy with it. The traditional software development methods are dependent on a set of predetermined processes and on-going documentation which is written as the work progresses and guides further development. The success of a project which is approached in this way relies on knowing all of the requirements before development begin and means that implementing change during the development lifecycle can be somewhat problematic. However, it also makes it easier to determine the costs of the project, set a schedule and allocate resources accordingly.

C. Characteristics of Agile Approach

Agile is for People

The most important implication to managers working in the agile manner is that it places more emphasis on people factors in the project: amicability, talent, skill, and communication. These qualities become a primary concern for the would-be agile team. Skill development is important, so that each person can deliver more value with time. The attention to the human issues gives agile projects a particular feel. It is not just for reducing paper work and hoping that everything else will fall into place. To paraphrase one developer,

"A small, rigorous methodology may *look* the same as an agile methodology, but it won't *feel* the same."

The agile group concept grows to span teams, organizations, and other working relationships. It is very difficult for agile people to function well in a rigid organization - and vice versa.

Agile processes are designed to capitalize on each individual and each team's unique strengths.

Team Competence

Agile teams are characterized by self organization and intense collaboration, within and across organizational boundaries. Self-organizing teams are not leaderless teams; they are teams that can organize again and again, in various configurations, to meet challenges as they arise. Agility requires that teams have a common focus, mutual trust, and respect; a collaborative, but speedy, decision-making process; and the ability to deal with ambiguity. We

distinguish collaboration from communication. Communication is sending and receiving information. Collaboration is actively working together to deliver a work product or make a decision. To identify the differences between traditional and agile project managers, one of which is: "A traditional project manager will normally focus on agreeing to a detailed contract with customers about the totality of the system to be delivered along with the costs and timescales".

Agile Organizations

An agile team working within a rigid organization has as difficult a time as agile individuals working within a rigid team. Many project teams we have interviewed report that when they responded to customers' requests and to technology platform changes to deliver usable value to the customer, their management admonished them for lack of conformance to the original plan. Agile organizations and agile managers understand that demanding certainty in the face of uncertainty is dysfunctional. Agile companies practice leadership-collaboration rather than command-control management. They set goals and constraints, providing boundaries within which innovation can flour-focused on setting up a collaborative relationship with the customers." They are macromanagers rather than micromanagers. They understand that who makes decisions isn't as important as collaboration on information to make informed decisions. They understand that agilitydepends on trusting individuals to apply their competency in effective ways.

D. The Agile Alliance

In early 2001[16], motivated by the observation that software teams in many corporations were stuck in a quagmire of ever increasing process, a group of industry experts met to outline the values and principles that would allow software teams to develop quickly and respond to change. They called themselves the Agile Alliance. Over two days they worked to create a statement of values. The result was the manifesto of the Agile Alliance. Over the next three months they continued to work together to create the principles of agility.

Individuals and interactions over processes and tools

People are the most important ingredient of success. A good process will not save the project from failure if the team doesn't have strong players; but a bad process can make even the strongest of players ineffective. Even a group of strong players can fail badly if they don't work as a team. A strong player is not necessarily an ace programmer. A strong player may be an average programmer, but someone who works well with others. Working well with others, communicating and interacting, is more important that raw programming talent. A team of average programmers who communicate well are more likely to succeed than a group of superstars who fail to interact as a team. The right tool is very important to success. Compilers, IDEs, source code control systems, etc., are all vital to the proper functioning of a team of developers. However, tools can be overemphasized. An overabundance of big unwieldy tools is just as bad as a lack of tools. It should not be assumed that outgrowing a tool until it is tried and is not used. Instead of buying the top of the line, mega-expensive, source code control system, find a free one and use it until it is demonstrated and outgrown. Before buying team licenses for the best of all CASE tools, use white boards and graph paper. Before committing to the top-shelf behemoth database system, try flat files.

Working software over comprehensive documentation

Software without documentation is a disaster. Code is not the ideal medium for communicating the rationale and structure of a system. Rather, the team needs to produce human readable documents that describe the system, and the rationale for their design decision. Too much documentation is worse than too little. Huge software documents take a great deal of time to produce, and even more time to keep in sync with the code. If they are not kept in sync, then they turn into huge lies, and become a significant source of misdirection. A new team member is trained for the system by closely working with them and by transferring knowledge by sitting next to them and helping them. New team members are made part of the team through close training and interaction.

The two documents that are the best at transferring information to new team members are the code and the team. The code does not lie about what it does. It may be hard to extract rationale and intent from the code; but the code is the only unambiguous source of information. The team holds the ever-changing roadmap of the system in their heads. There is no way to put that roadmap down on paper and transferit to others that is faster and more efficient than interaction with the team.

Many teams have gotten hung up in pursuit of documentation instead of software. This is often a fatal flaw. There is a simple rule that prevents it:

Produce no document unless its need is immediate and significant.

Customer collaboration over contract negotiation
Software cannot be ordered like a commodity. Description of the software that is needed cannot be written and then can someone develop it on a fixed schedule for a fixed price. Sometimes software projects in this manner fails and the failures are spectacular. It is the duty of the managers of a company to tell their development staff what their needs are, and then expect that staff to go away for awhile and return with a system that satisfies their needs. But this mode of operation leads to poor quality and failure. Successful projects involve customer feedback on a regular and frequent basis. Rather than depending upon a contract, or a statement of work, the customer of the software works closely with the development team, providing frequent feedback on their efforts. A contract that specifies the requirements, schedule, and cost of a project is fundamentally flawed. In most cases the terms it specifies become meaningless long before the project is complete. The best contracts are those that govern the way the development team and the customer will work together.

Responding to change over following a plan
It is the ability to respond to change that often determines the success or failure of a software project. When we build plans, we need to make sure that our plans are flexible and ready to adapt to changes in the business and technology. The course of a software project cannot be predicted far into the future. There are too many variables to account for. We simply aren't very good at estimating the cost of a large project. The business environment that the software must serve is likely to change during the course of development. It is difficult to write reliable requirements. Customers are likely to alter the requirements once they see the system start to function. It is tempting for novice managers to create a nice PERT or Ghant chart of the whole project, and tape it to the wall. They may feel that this chart gives them control over the project. They can track the individual tasks and cross them off the chart as they are completed. They can compare the actual dates with the planned dates on the chart and react to any discrepancies. But what really happens is that the structure of the chart degrades. As the team gains knowledge about the system, and as the customer gains knowledge about their needs, certain tasks on the chart will become unnecessary. Other tasks will be discovered and will need to be added. In short, the plan will undergo changes in *shape*, not just changes in dates.

II. RELATED WORK

For our study, we have reviewed several papers on Open Source Development and agile and traditional approaches to see the details of the prior work done in this area.

Yu BengLeau, WooiKhong Loo, Wai Yip Tham and Soo Fun Tan**[1]** differentiated between agile and traditional approaches. It depicts the necessary phases in software development. It also suggests improvements for current agile development so that this lightweight SDLC could be adopted more in practice for organizational project management.

JesperHolck, Roberto Zicari, and Volker Mahnke in **[2]** highlight the key elements in success of business and further explained about the open source software requirement in business.

Michael W. Godfrey and QiangTu**[3]** explains about the early phases for open source software as the evolution of open source.. They have found that Linux had been growing at a super linear rate for several years.

Carolyn A. Kenwood in **[4]**, illustrates the business case of open source software particularly military department of USA. It was concluded that OSS is a viable long-term solution for products relevant and interesting to a large community with highly skilled developers.

Mikio Aoyama in **[5]**, introduces ASP (Agile Software Process) as a new software process model. The model aims at quick delivery and is time based. The feedback factor gave higher quality and customer satisfaction. It was concluded that ASP model suitable for distributed concurrent development environment.

ChituOkoli and Kevin Carillo**[6]**,explained that Open source software development (OSSD) is a promising alternative for synthesizing agile and plan-driven (e.g. waterfall) software development methodologies that retains most benefits of the two approaches. Comparison of the two approaches with OSSD is done.

Livermore in **[7]** presented a survey based research project which investigated about agile software development methodologies using iterative development, prototyping templates and minimal documentation requirements. The research project investigated agile SDM implementation using an online survey sent to software development practitioners worldwide.

AlenaBuchalcevova in **[8]** presents a survey to determine the rate of agile approaches usage and practical experience with these approaches in software houses of Czech Republic. The analysis of this survey showed that the use of agile methodologies and approaches are at a nascent stage and its usage (as indicated by the survey) will be rising as agile approaches mature. It was concluded that Agile methodologies have matured and have been scaling along a number of dimensions geographic distribution and global development, number of collaborations with suppliers, combined hardware/software projects including and beyond embedded software, team size, project size, mission criticality, and involvement with legacy systems.

G.I.U.S. Perera in **[9]** states that Agile software process possesses standard characteristics of a process paradigm. A study is conducted on a student programming project and the results showed Student's Skill Improvement. The study took a different approach from the rest of the Agile process based studies on student learning, by focusing more on the individual student's knowledge improvement through the practice.

Bea During &HolgerKrekel**[10]** represented different aspects of agility within the open-source.. It was concluded that the cumulative effects of an agile, open and dynamic team process combined with a market and curious first adopters facilitates agile business.

T. J. Halloran, William L. Scherlis in **[11]** conducted a preliminary survey of a number of open source projects to gain a clearer understanding of the practices used by active and successful open source projects. The surveyed projects focus on peopleas the locus of Quality Assurance (QA). Clearly, responsibility for quality rests on those who have code commit privileges.

Sulayman K. Sowe, IoannisStamelos, Lefteris Angelis **[12]** stated that Free/Open Source software projects are people oriented. It was concluded that the posting and replying activities of the participants and knowledge is externalized and internalized into each lists, the trend in knowledge sharing in the lists, the correlation between posting and replying activities, and the self-organizing nature of the individual's knowledge sharing activities.

Richard Vidgen, Tredor Wood-Harper & Robert(JRG) Wood in **[13]**, presented traditional approaches to Information System have concentrated upon a production view of quality associated with a controlled development process and metrics that monitor attributes such as software usability, the number of software errors, and developer productivity.Soft Systems Methodology is a considered as relevant notion of Information System use quality.

Murat Yilmaz, Rory V. O'Connor and Paul Clarke in **[14]** stated that the vision of building a successful software product requires teams of individuals equipped with a wide range of social and technical The data from a survey conducted on 266 software practitioners to ascertain job roles in two middle size software companies, one of which uses traditional methods while other uses a tailored agile methodology.

Ambily O. A., Dr. T. JudethMalliga in **[15]** discussed that the processors are getting faster, hard disks becoming compact, bandwidth increasing, operating systems & programming tools evolving and security is increasing as well. It was concluded that agile methods are a group of development approaches to software development.

III. RESEARCH METHODOLOGY

The research is conducted with the different employees of different companies. The survey is done to see whether the companies are following agile practices or traditional practices for software development.To study the nature of Open source development strategies being adopted by the Software companies with degree of the agility of the Software Delivery so that we can analyze the adoption of open source software.

Objectives:

To analyze the nature of open source software development in Indian industry and to analyze whether the companies are following agile practices or traditional practices for software development.

Research Design: Research design provides a blueprint for a study. Descriptive and exploratory research design will be followed.

Sampling Design: For the purpose of data collection, the whole Open source Industry will be divided into different clusters.
A well structured questionnaire is prepared to obtain responses from target developers.
Sample Size: 8
Sample units: Software development team.
Sampling Technique: Random Sampling/Cluster Sampling.

The whole set of data will be collected through questionnaires by taking samples and forming clusters of complete population (Industry). Within each cluster, random sampling will be used to take responses from software developers.

Data Collection and Analysis: Based on the objectives of the study, data is collected through both primary and secondary data collection method. To obtain responses through questionnaire, a survey is conducted through well designed questionnaires which will be presented to software developers for their responses. However, validity and reliability of the questionnaire will be judged through initial pilot survey and further existing statistical tests will be applied. To quantify the responses, depending upon the need of study likert scale (5 point) technique will be used and analysis of data will be done using various statistical techniques. Based on the requirement of collected data Chi-square test is applied to validate the results of our research.

Parameters for Performance Analysis:
The SPSS (Statistical Package for social sciences) software is used for evaluating the alpha value of the questionnaire. A questionnaire that is said to be good has an alpha value of 0.6 and based on this questionnaire we calculated the alpha value and it came out to be 0.8 which is consider very good. SPSS is a comprehensive and flexible statistical analysis and data management solution. SPSS can take data from almost any type of file and use them to generate tabulated reports, charts, and plots of distributions and trends, descriptive statistics, and conduct complex statistical analyses ofpredictive calculation used to determine the relative effect of a single factor on a situation.

IV. EXPERIMENTAL RESULTS

Structured questionnaire are used to collect data from different information technology companies. The questionnaire is prepared by using the seven point likert scale. Interviews of respondents are also conducted for the collection of information regarding Open source adoption in prospective of agile techniques.

Questionnaire:
To achieve the set of objectives, survey questionnaire are presented to the respondents with the criteria to assess the knowledge sharing techniques in organizations. Questionnaires are filled by the various employees from various departments of the companies.
The questionnaire is based upon elements of existing Open source sharing structure. It is divided into four parts:

1. General information:- This part of questionnaire consists four questions to collect information regarding potential open source sharing policies.
2. Current situation:- This part of questionnaire consists nineteen questions about the use of current methodologies.
3. Gaps and Desiderata:- This part of questionnaire consists three questions about to identify gaps in existing techniques.
   Future lines of research:- this part of questionnaire consists six questions about the adaption of new techniques.

Likert scale:
Likert scale is a psychometric scale commonly used in questionnaire based research. It is used the likert items. It is available in different sizes like five points likert scale, seven points likert scale etc. Questionnaire prepared with the help of likert scale is easy to analyze. It does not expect a direct yes or no from the respondent, but it allow for the degree of opinion. In this study a standardized questionnaire are used to measure the impact of Open source development in distributed agile environment. This questionnaire seven point likert scale is used. The pre-coded responses offered by this are given below:

1. Strongly disagree

2. Disagree
3. Disagree somewhat
4. Undecided
5. Agree somewhat
6. Agree
7. Strongly agree

These responses have the numericalvalues from 1-7 that are used to measure the attitude under investigation.

Reliability test

The consistency of a score from a measurement scale is assessed with reliability test. There are some statistical methods are available which are used to check the reliability. In this research for checking the reliability of questionnaire the Cronbach's alpha is used. The value of Cronbach's alpha is lies between 0 and 1. A higher value of this shows the greater consistency in variance of the sample test score. Usually the value of Cronbach's alpha is more than 0.6 is considered to be reliable in survey research, but some statisticians considered 0.7 to be reliable.

Reliability Statistics

Cronbach's Alpha	N of Items
.888	33

From the above test we find the Cronbach's alpha is .888, which is more than 0.7 (.888>0.7). Hence we can say that the questionnaire formed is reliable.

Analysis of Questionnaire:

Assumptions 7 rates the highest priority and 1 rates as lowest so as less as the mean more is the better solution. If Mean value is lies in between 3.5 to 1.5 then the companies use the Extreme programming methodology, otherwise Adaptive software development is used. Zero response is invalid and cannot be considered as part of validation.

A Sample size of 10 companies was selected, out of which only 8 companies given the response of the questionnaire.

Test of hypothesis:

For checking whether companies are using traditional approach or distributed agile approach hypothesis is set. In this study chi-square test is used to check the acceptance of hypothesis.

In chi- square test we use the highest value from strongly agree, agree, and agree somewhat for agile and, highest value from disagree somewhat, disagree and strongly disagree is used for traditional approach.

The hypothesis used in this study is given below:

H(0): Companies are working under traditional approach for open source development.

H(1): Companies are working under distributed Agile approach for open source development.

The result of chi-square test is given below: The result of chi-square test is given below:

Variable	Agile	Traditional	Total
Company 1	40	20	60
Company2	33	60	93
Total	73	80	153

Chi-Square calculations:

=153[40*60-20*33]^2/ 60*93*73*80

=153[1740]^2/ 32587200

=153[3027600]/ 32587200

=463222800/32587200

X^2 = 14.21

Df= (2-1)*(2-1)= 1

Alpha level of significance: 0.5

probability level (alpha)

Df	0.5	0.10	0.05	0.02	0.01	0.001
1	0.455	2.706	3.841	5.412	6.635	10.827
2	1.386	4.605	5.991	7.824	9.210	13.815
3	2.366	6.251	7.815	9.837	11.345	16.268
4	3.357	7.779	9.488	11.668	13.277	18.465
5	4.351	9.236	11.070	13.388	15.086	20.517

We now have our chi square statistic ($x^2 = 14.21$, our predetermined alpha level of significance (0.5), and our degrees of freedom (df = 1). Entering the Chi square distribution table with 1 degree of freedom and reading along the row we find our value of x^2 (14.21) is bigger than value defined in table. The corresponding probability is above 0.001 probability levels. That means that the p-value is below 0.5. Since a p-value is greater than the conventionally accepted significance level of 0.5 (i.e. $p < 0.05$) we will reject the null hypothesis. In other words, Companies are working under distributed Agile approach is in use.

Variable	Agile	Traditional	Total
Company 3	22	18	40
Company4	38	20	58
Total	60	38	98

Chi-Square calculations:
= 98[22*20-18*38]^2/ 40*58*60*3
=98[-244]^2/ 5289600
=98[59536]/ 5289600
=5834528/5289600
X^2 = 1.103
Df= (2-1)*(2-1)= 1
Alpha level of significance: 0.5

probability level (alpha)

Df	0.5	0.10	0.05	0.02	0.01	0.001
1	0.455	2.706	3.841	5.412	6.635	10.827
2	1.386	4.605	5.991	7.824	9.210	13.815
3	2.366	6.251	7.815	9.837	11.345	16.268
4	3.357	7.779	9.488	11.668	13.277	18.465
5	4.351	9.236	11.070	13.388	15.086	20.517

We now have our chi square statistic ($x^2 = 1.103$, our predetermined alpha level of significance (0.5), and our degrees of freedom (df = 1). Entering the Chi square distribution table with 1 degree of freedom and reading along the row we find our value of x^2 (1.103) is less than value defined in table. The corresponding probability is below 0.10 probability levels. That means that the p-value is above 0.5. Since a p-value is lesser than the conventionally accepted significance level of 0.5 (i.e. $p > 0.05$) we fail to reject the null hypothesis. In other words, Companies are working under traditional approach and knowledge sharing is not in use.

Variable	Agile	Traditional	Total
Company 5	38	0	38

Company 6	40	10	50
Total	78	10	88

Chi-Square calculations:
=88[38*10-40*0]^2/ 38*50*78*10
=88[380]^2/ 1482002
=153[144400]/ 1482000

X^2 = 8.574
Df= (2-1)*(2-1)= 1
Alpha level of significance: 0.5

probability level (alpha)

Df	0.5	0.10	0.05	0.02	0.01	0.001
1	0.455	2.706	3.841	5.412	6.635	10.827
2	1.386	4.605	5.991	7.824	9.210	13.815
3	2.366	6.251	7.815	9.837	11.345	16.268
4	3.357	7.779	9.488	11.668	13.277	18.465
5	4.351	9.236	11.070	13.388	15.086	20.517

We now have our chi square statistic ($x^2 = 8.574$), our predetermined alpha level of significance (0.5), and our degrees of freedom (df = 1). Entering the Chi square distribution table with 1 degree of freedom and reading along the row we find our value of x^2 (8.574) is bigger than value defined in table. The corresponding probability is above 0.01 probability levels. That means that the p-value is below 0.5. Since a p-value is greater than the conventionally accepted significance level of 0.5 (i.e. $p < 0.05$) we will reject the null hypothesis. In other words, Companies are working under distributed Agile approach is in use.

Variable	Agile	Traditional	Total
Company 7	40	12	52
Company 8	36	5	41
Total	76	17	93

Chi-Square calculations:
=93[40*5-12*36]^2/ 52*41*76*17
=93[-232]^2/ 2754544
=93[53824]/ 2754544
=5005632/2754544
X^2 = 1.817
Df= (2-1)*(2-1)= 1
Alpha level of significance: 0.5

probability level (alpha)

Df	0.5	0.10	0.05	0.02	0.01	0.001
1	0.455	2.706	3.841	5.412	6.635	10.827
2	1.386	4.605	5.991	7.824	9.210	13.815
3	2.366	6.251	7.815	9.837	11.345	16.268
4	3.357	7.779	9.488	11.668	13.277	18.465
5	4.351	9.236	11.070	13.388	15.086	20.517

We now have our chi square statistic ($x^2 = 1.817$, our predetermined alpha level of significance (0.5), and our degrees of freedom (df = 1). Entering the Chi square distribution table with 1 degree of freedom and reading along the row we find our value of x^2 (1.817) is less than value defined in table. The corresponding probability is below 0.10 probability levels. That means that the p-value is above 0.5. Since a p-value is lesser than the conventionally accepted significance level of 0.5 (i.e. $p > 0.05$) we fail to reject the null hypothesis. In other words, Companies are working under traditional approach.

V. CONCLUSION

Agile methodologies in an organization play an important role for the success of projects. According to the methodologiesexplained reflect that agile methodologies provide better relation, customer satisfaction and quality at all the phases of software development life cycle. There is mix behavior in using the Agile and traditional methodologies in the industry as suggested by our analysis. As some companies usually have some particular project which can be done in traditional methodologies and some companies have development projects which tend to fit agile methodologies more.

REFERENCES

[1] Yu BengLeau ,WooiKhong Loo, Wai Yip Tham and Soo Fun Tan, "Software Development Life Cycle AGILE vs Traditional Approaches"2012 International Conference on Information and Network Technology (ICINT 2012) IPCSIT vol. 37 (2012) © (2012) IACSIT Press, Singapore, 2012.

[2] JesperHolck, Roberto Zicari, and Volker Mahnke, "A Framework Analysis of Business Models for Open Source Software", Copenhagen Business School, Department of Informatics, Howitzvej 60, DK-2000 Frederiksberg, Denmark, 2000.

[3] Michael W. Godfrey and QiangTu, "Evolution in Open Source Software: A Case Study", Software Architecture Group (SWAG) Department of Computer Science, University of Waterloo.

[4] Carolyn A. Kenwood, "A Business Case Study of Open Source Software", Contract No.: DAAB07-01-C-C201, Project No.: 0700M520-AA, MitreWashington C3 Center Bedford, Massachusetts.

[5] M. Aoyama, "Agile Software Process and its Experience", in proceedings of 20th ICSE, Kyoto, Japan, pp. 3-12, ISBN: 0-8186-8368-6, 1998.

[6] ChituOkoli and Kevin Carillo," The best of adaptive and predictive methodologies: Open source software development, a balance between agility and discipline" (c) 2010.

[7] Livermore, A. Jeffrey, "Factors that Significantly Impact the Implementation of an Agile Software development Methodology", Journal of Software, Vol. 3 (4), pp. 31-36, 2008.

[8] A. Buchalcevova, "Research of the Use of Agile Methodologies in the Czech Republic", C. Barry, M. Lang, W. Wojtkowski, G. Wojtkowski, S. Wrycza, &Zupancic, The InterNetworked World: ISD Theory, Practice, and Education. Springer-Verlag: New York, ISBN 978-0387304038, 2008.

[9] G.I.U.S. Perera, "Impact of using agile practice for student software projects in computer science education", International Journal of Education and Development using Information and Communication Technology (IJEDICT), 2009, Vol. 5, Issue 3, pp. 85-100.

[10] Bea During, HolgerKrekel, "Open Source, EU funding and Agile Methods"

[11] T. J. Halloran, William L. Scherlis," High Quality and Open Source Software Practices".

[12] Sulayman K. Sowe *, IoannisStamelos, Lefteris Angelis," Understanding knowledge Sharing activities in free/open source software projects: An empirical study" The Journal of Systems and Software - (2007)

[13] Richard Vidgen, Trevor Wood-Harper & Robert Wood," A Soft Systems Approach to Information Systems Quality", Scandinavian Journal of Information Systems, Vol. 5, pp. 97-112, 1993.

[14] Murat Yilmaz, Rory V. O'Connor and Paul Clarke, "A Systematic Approach to the Comparison of Roles in the Software Development Processes"

[15] Ambily O. A., Dr. T. JudethMalliga, "Agile Software Development anApproach to Light Weight From Heavy Weight", Ambily O.A Et Al. / International Journal of Engineering Science and Technology (Ijest)

[16] Heinrich Heine,"Agile Processes".