IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661 Volume 4, Issue 4 (Sep-Oct. 2012), PP 13-23
www.iosrjournals.org

Technology Integration Pattern For Distributed Scrum of Scrum

Madiha Kanwal[1], Muhammad Umer Sarwar[2],Usman Ali Sarwar[3]
Shazia Riaz[4],Muhammad Irfan Khan[5]
[1]*(College of Computer Science& Information Studies, GC University, Faisalabad, Pakistan,*
[2]*(Assistant Professor (CS), College of Computer Science& Information Studies,*
GC University, Faisalabad, Pakistan
[3]*(Computer Science Department, Agriculture University, Faisalabad, Pakistan,*
[4]*(College of Computer Science& Information Studies, GC University, Faisalabad, Pakistan,*
5(Lecturer (CS), College of Computer Science& Information Studies, GC University, Faisalabad, Pakistan,

Abstract: *For distributed project development, the main problem is the handling of the integration of decomposed components so that less problems. Integration problems are the major cause of failure for distributed project development. In past lot of work is done for integration of distributed componentswhen scrum process is implemented for development but one important factor which is ignored is the technology factor, it may be possible that decomposed components may run on different technologies. Decomposed components functionality may dependent on one another, so technology of one component affect the functionality of other components. So according to our research work there is a need to consider technology as an important factor during distributed development. To handle this problem we purposed a new pattern that controls the Technology Integration forDecomposedComponents for Scrumprojects.We found better result in fact 80 percentage more chances of project survival by implemented this pattern. This pattern increases the project maintainability by reducing the risk factor;increases project scalability.This pattern produces positive affect on project life and development. We also investigate the previous Pattern implemented on Distributed Scrum of Scrum.*

Keywords: *Agile Scrum, Distributed Scrum of Scrum, Integration Pattern, Technology,*

I. Introduction

For distributed team projects, communication among team is the major cause of letdown. Communication directly above is often presented by the way development is assigned to separate development teams and a discrepancy in the functionalities necessary for a subsystem. In distributed team development, a large degree of collaboration between team members is required; as some part of system is developed by one team is subject to the service station of other parts developed by teams scattered at separate places.

In history lot of work is done to solve many problems related to the distributed team development. From literature review and our research work it is clearly identified that in past no work was done for distributed scrum related to technology. It is also identified that if development teams does not handle technology factor properly, then it creates problems for team members/teams at the last stage of development when these components are integrated to shape complete software, which in turn cause the unstabilization of complete project and failure of the project and also wastage lot of time.

For decomposed distributed project, the main problem is the handling of the integration of decomposed parts so that less problems for the complete software execution after development. In history researchers do work for integration of distributed parts but one important factor which is ignored is the technology factor because for software projects, it may be possible that decomposed parts may run on different technologies (e.g. decomposed parts may run on different hardware platforms). Decomposed components functionality may dependent on other components, so technology of one component effect the other component and there is a need to consider technology as an important factor.

Agile processes are simple, iterative, which emphasize creativity and collaboration. The agile development methods have grew from the well-known and old iterative and incremental methodologies. These approaches were grounded on the belief that modification and innovation would yield better outcomes and this belief is the base of human reality and the product development culture.

Agile principles (Table 1) focused on producing the software early and quickly that customer can get on hands, instead of writing detailed specifications and spending lot of times on these specifications. Agile development emphasis on cross-functional teams which have ability to made decisions. Opposite to other development methods, in agile development method the team hierarchy is not strict. The team of agile

development is generally self-organizing and multi-dimensioning. In contrast to general conventional methods, these methods make sure better resolution for software enhancement. From survey investigation it was revealed that agile technique produced value-added result for software development methods in many aspects for example quality of software, final form of software product inside almost equal cost, reliability of user, quick response to user and opportunity of enhancement (Boehm, 2002).
In agile practice things are slightly planned and consists of small increments to done these things. In contrast to traditional methods, agile systems are less amounting of documented. Agile procedures more focused on code and more emphasis on working product. Agile methods are adaptive way to develop and quickly implement software with close

1.1. **Scrum**

Among agile methods, scrum is becoming a popular method for project controlling. Scrum is more focused on how to manage the development process just like a framework and states nothing about how to develop software (e.g. unlike eXtreme Programming). For the team that develops software system, agile software development processes like Scrum that is more focused and aimed to enhance transparency, energy and clarity to the (Table 2). In Scrum, no need complete requirements specification before development start because Scrum provides software earlier. There are a lot of methods to implemented Scrum process on running system (Cho, 2008). During Scrum, software development phases run parallel (etc. investigation, description of requirements, design, operation and testing).

Scrum involves one or many self-managing, cross-functional, teams (consists of about seven members) to deliver incremental product and it is just like a management framework. Scrum teams used fixed length repetitions, these repetitions are called Sprints and usually the size of Sprint varies from two weeks or 30 days long.

The selected items do not modify during Sprint. Every day the team meets for a short time to report the update and progress to each and other with the help of charts description that turn to the remaining effort. The development team analyses Sprint with shareholders and determines what they have built at the end of Sprint. Feedback obtained from people and that feedback can be integrated into the next Sprint. At the last of every Sprint, Scrum more focused on working product.

1.1.1 **Roles**

Scrum Master
Product Owner
Team

Table 1: Main Beliefs of Agile Manifesto

Sr No.	**Main Beliefs**
1	The main concern of agile is the early and constant delivery of functioning software to satisfy the customer.
2	It makes possible modification in requirements, even later in development phases.
3	The main priority is to deliver working software frequently within the short time period.
4	Both community (creators and business persons) works with each other for the whole project.
5	Motivated people shape the project. These methods provide them fully supported environment and trust them.
6	Discussion among end users and developers is the best approach of information conduction.
7	Progress of the project is measured with working software.
8	Agile encourages maintainable development process. The developers, benefactors and end users must be capable to retain a continuous step for an indefinite period.
9	Agile improves design with constant devotion to technical excellence.
10	Simplicity is the base of exploiting the amount of work not done is important.
11	It implemented self-organizing team arises best architectures for requirements and designs. The team regulates its conduct analysis procedure of in what way to turn out to be more in effect.

Table 2: Effect of Scrum on Project

Project Characteristics	**Scrum**

Primary objective	Quick Response
Requirements	Mostly Unknown, Rapid Variation, Up-and-coming,
Risks	Unknown Risks, Major Impact
Architecture	Considered for Current Necessities
Size	Small Teams and Decomposition of Tasks
Customers	Knowledgeable, Cooperated and Committed
Planning and management	Qualitative Mechanism, Coopted Plans
Refactoring	Inexpensive
Developers	Agile, Cooperated and Knowledgeable

1.1.2. Artifacts

Sprint Backlog
Product Backlog
Burndown Chart

1.1.3. Ceremonies

Sprint Review
Sprint Planning
Daily Scrum Meeting

1.2. Cross Layout Scrum of Scrum

For projects which are distributed crossways geographies, in such type of projects teams work separated physical locations and these teams work on different components that are inter-dependent and these components together make up the application, these teams may have their own Scrums at their own location. However, these teams get together, share important points, coordinate and also resolve inter-dependencies. For such type of circumstances, it is vital to hold a daily Scrum of Scrums to create the coordination between teams to make them more efficient because of the impediment of different working styles and cross-geographies.
There is different type of dependencies for Scrum of Scrum that effect the coordination between the teams.

1.3. Organizational Patterns

Agile development entangled with organizational patterns from start directed work of the Coplien paper (interactive pattern reviewer) and highlighted the effect of this work on XP (Fraser et al,2003). Recently, persons related to Scrum have taken attention with organizational patterns (Sutherand, 2008).
Organizational culture is the pattern adaptation of basic conventions that a given group has designed, revealed, or developed in learning to handle with harms of internal integration and outside edition plus that work have been sound sufficient to be measured effective.

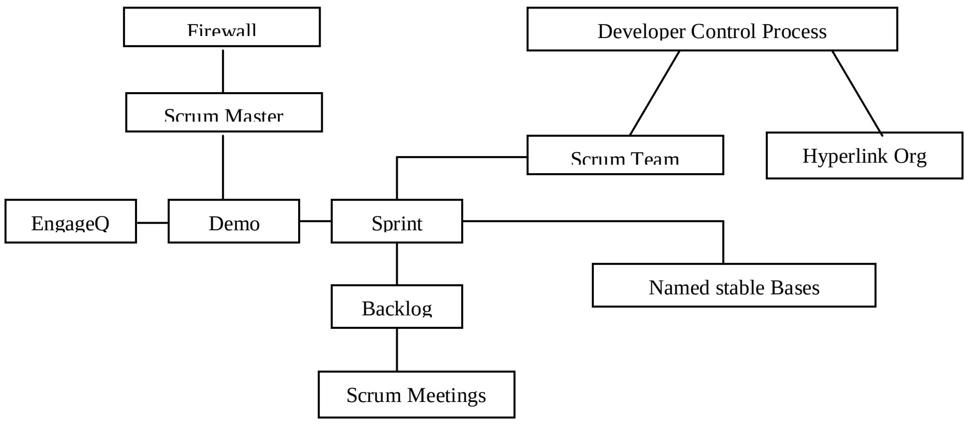

Figure 1: Scrum Pattern Language

Organizational patterns are solutions to common software development problems by a current body of literature. This literature has gained importance as the software projects become more complex related to organizational matters. These patterns describe more about the structure of organizations rather than the construction of software and adapting pattern discipline is a normal.

Organizational patterns bond with the existing software pattern. The pattern describes anthropology and the general ground area of organization and process.

The above given figure described the associations between the organization and Scrum patterns (Fig 1).

For organization software development process formation, Scrum agile method can be combined with organization pattern (Coplienand Schmidt, 1995). These patterns are decomposed into four types of languages.

II. Previous Research

Schwaber (1995) stated that accepted philosophy related to development method has proved incorrect; according to this method the development methodology can be scheduled, predictable and effectively finished for system. Opposite to the traditional methods Scrum adopts the approach is that the development procedure can only be approximately description of the overall development process and the process is changeable and complex. He stated that the Scrum consists of group of movable actions and these actions gathered which are called workable products and Scrum team consist of best professionals that implement best practices to develop systems. Since the Scrum consists of loose activities, so more chances of risks and there is a need of management to handle process. Scrum is an improvement to the incremental object-oriented process.

Dent (1996) implemented the early review method in which use of documentation to review the work progress. This method implementation showed positive results and also shows successful implementation in individual and small teams. The management main focused on the handling of risk problems during Scrum process. This method helps to give value to individual developers and constitutes big teams for software engineering.

Beedle (2000) proved that during whole Scrum process sharing of information among developers so created a knowledge process. The main features of Scrum is the commitment of project completion date, arranging functionality according to the importance, recognizing accessible assets for process and preparing important assessments related architecture. For Scrum it is assumed that requirements are changeable so the time period of planning phase is small in contrast to traditional methods. During process the change, constant sharing of information and response back increases performance and belief. Its implementation created atmosphere which consists of definite parts and relations.

Sutherland (2001) provided a short reflective on Scrum bass, progress of Scrum in five companies. Provided an orientation for future study. Supported to verify Agile principles for Scrum etc. scalable, lightweight development method. It is said that Scrum can be positively work in every environment by the results of implementing Scrum in five organizations and these organizations are of different sizes and implement different technologies

Cho (2002) claimed that agile scenario methods have been established and grew meanwhile initial of 1990s. Agile procedures extensively used in business environment due to the small iterative and incremental development rotation. It is best suited for business environment due to changes in requirements. This paper will helpful for Scrum (agile) implementation in organization. He also presented more famous roles, services and relics of agile for industry. He identified five problems like documentation, cooperation among developers, customer participation and working atmosphere.

Potter and Sakry (2003)showed that Scrum has better results for organizations of Level 2. Then it is possible to use Scrum and CMMI collectively. All the left over performs for Levels 2 and 3 can be applied with Scrum.

Sutherland (2004) started Scrum at Easel Corporation where the first Object-Oriented Analysis and Design (OOAD) tool was constructed. Graphic design tool generated auto-code in Smalltalk development atmosphere and modifications in code were directly reproduced back in design.

Sutherland et al (2005) provided a brief overview of a Continuous Flow Scrum when it was originally published. A more profound insight into these issues can be found through study of set-based concurrent engineering.

Barton et al (2005) stated that agile methods give better chance to produce good software for business environment. Business peoples mainly interested in Scrum due to its main focused on ROI and fast application. It is showed that agile is better than conventional methods due to refining the reportage ability. It was presented overview for the discussion and participation of different peoples related to Scrum. For project progress the main focused on visibility of project to executive management.

Sutherland (2006) claimed that the main aim of Scrum was to improve efficiency and decreased time due to market demand. He created All-At-Once Scrum by implementing numerous overlying sprints inside the same Scrum teams. This practice provides increasing application functionality over opponents. An implementation of All-At-Once Scrum showed in what way mobile/wireless teams get revenue benefit in 2007 with the Scrum automation started in 2000. While All-At-Once Scrum is not for starters instead it produced good and quick results for professional implementation.

Striebeck (2006) Showed that skill agile leaders made it possible to implement agile in Google.Google where it does not possible to have empathy to procedures in general.

Qumer and Henderson (2006) with the help of framework valuation tool (4-DAT) matched two agile development approaches (Scrum and XP). For estimate purpose, they implemented both computable and qualitative methods and also they implemented at both the stage level and the preparation level. It is concluded from this analysis that in XP more agile implementation at phase level and less agile implementation at practice level than Scrum. This assessment tool helped the manager and developers to choice suitable agile method according to the organization and specific project.

Sutherland et al (2007) analyzed and recommended suitable practices for distributed teams structure. This study verified that distributed (even outsourced) team structure can be creative as a small collected team. For this implementation required good engineering performs alongside with Scrum. The complete group of teams must work as a solo team with the help of build fountain, tracking mechanism.

Jakobsen (2008) presented particularly how agile procedure (Scrum) effectively joint with CMMI. CMMI basically consists of disciplines to reflect on. With the implementation of these disciplines created a center of attention on chief aspects for agile procedures (e.g. quality of product backlog). The combined Scrum with CMMI disciplines have lead organized method to run Scrum normal actions and produce better results. This twist of Scrum and CMMI produce fruitful outcomes.

Blagojevi*et al.* (2008)recognized four diverse models that implemented Scrum with respect of two aspect: team's collective dedication and problem resolving capability. During this work, the applicability of Scrum is evaluated in the perspective of hardware/software codevelopment, services for progress, program management and scientific research. Two parameters are important to understand these expectations: the team's resolving ability and their level of shared commitment to sprint goals.

Sutherland*et al.* (2009) created a distributed environment for Scrum that produce as the same rate and quality as the team located in single place and this is implemented on many projects and gained better results-. This team approach lower costs and allow rising and declining team size without knowledge failure. This approach is greatly suggested for skilled agile teams.

Sutherland*et al.* (2010) showed better performance by combining CMMI with Scrum. This combined outcomes produce the same results as either use CMMI or Scrum single-handedly. With the implementation of Scrum projects gained improvement in production and quality in contrast to conventional processes. These fallout helped in ROI based conclusion. Scrum decreased work of many aspects (etc. defects, modify, total required work) to extend of 50 percentages. This work has outlined how Systematic adopted scrum and story based development into its CMMI processes inspired from a strategic focus on Lean. This work showed how general practices from CMMI can be implemented for agile methods.

Yaggahavita (2011) highlighted possible cultural impacts on scrum methodology when practiced in a cross cultural team. This research in its outcome produces a set of hypothesis that identified cultural behaviors that can have a probable impact on the scrum practices when practiced within a culturally distributed team.

III. Materials And Methods

In this part we described the patterns in the following mechanism:
The **context** implementation area of pattern
The **problem** pattern wants to resolve
The **forces** restriction on pattern implication
The **solution** of the pattern problem,
The **resulting context** that displays what are the outcomes if the solution is applied

3.1. Ignorance of Technology Factor

From the survey data it is cleared that technology factor is ignored by development team because their main focus is the functionality of the software. Developers ignore this factor because the customer wants the main output of a project is that it fulfills the required functions. This factor created more problems when scrum team is distributed and this team members/teams work on same project. Developers ignore this factor because the customer wants the main output of a project is that it fulfills the required functions. Almost 80% people ignore this factor, only 7% people consider technology and functionality equally important which is depicted in the Fig 2**.**

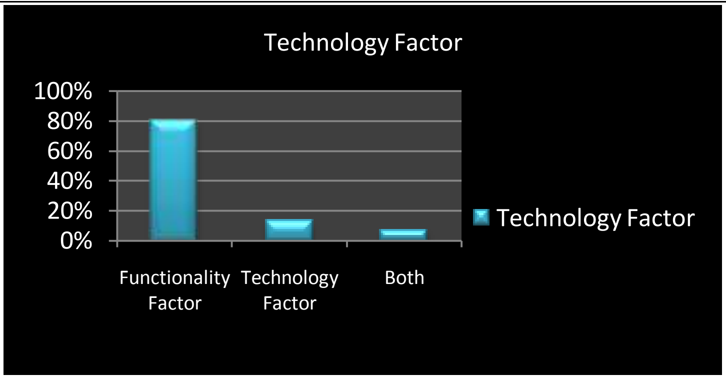

Figure 2: Ignorance of Technology Factor during Development

3.2.Integration of Technology

For decomposed distributed project, the main problem is the handling of the integration of decomposed parts so that less problems for the complete software execution after development. In history researchers do work for integration of distributed parts but one important factor which is ignored is the technology factor because for software project, it may be possible that decomposed parts may run on different technologies (e.g. decomposed parts may run on different hardware platforms). Decomposed subsystem functionality may dependent on other subsystem, so technology of one component affect the other component. So there is a need to consider technology as important as development team considered functionality. I purposed a pattern that controlled the integration of decompose parts with respect to technology point of view. In my work I suggested for distributed team a decomposition pattern. The project is decomposed into parts by Scrum Master during Sprint planning. Suggested decomposition of the project is done according to the two major aspects which are given in Fig 3.

3.3. Technology Decomposition and Integration Pattern

3.3.1. **Problem**

How to integrate the technology within the subsystem teams so that the system grew as a complete system. How to integrate the decomposed components so that system achieves described product functionality requirements with technology factor properly.

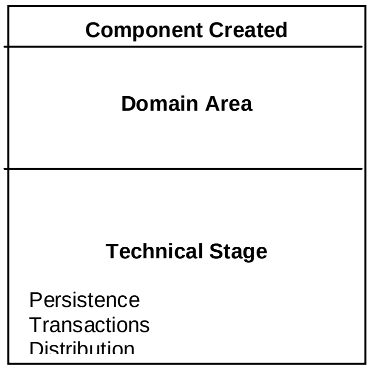

Figure 3: Define Technical Platform

3.3.2. Forces

- ✓ Technology necessities for the system may be imperfect and insufficient in the beginning
- ✓ Integrating technology components require configuration management tools plus testing burden
- ✓ The working product probably accessible at the finish stage of repetition. During iteration, the professionals conflict where devote their more effort, on the system evolution, or on integration for maintenance.
- ✓ In distributed team environment, an isolated team more focused on their assigned responsibilities related to subsystem. This may contract with the entire system goal
- ✓ Decomposed components may have functionalities dependencies
- ✓ Need of collaboration among teams due to work on same project
- ✓ For feedback purposes the work of teams must be incorporated on weekly basis. Software integration should happen very frequently decrease the problems of integration

3.3.3.Solutions

Every team kept an “Integrator manager” whose responsibility is to keep view the subsystem integration effort within his team and with other teams. When the project is decomposed into parts then scrum master decomposed according to the following category:

Category A: Subsystems that have functionality dependence and work on different technologies

Category B: Subsystems that have functionality dependence and work on compatible technologies

Category C: Subsystems that have no functionality dependence and work on compatible technologies

Category A Functionally dependent components must be assigned to one team or at least assigned to the teams that are near or within same time region. Establish a network for components a step-wise integration. This type of allocation allows integrating the components early as possible during Sprint cycle in to provide feedback to end user or customer. This early integration step wise technology integration secures technology stability for the working product. Development team tries to resolve technology integration problems at early level. It reduces the component technology compatibility problems and increases subsystem and overall project efficiency.

Category B These components may be allocated to the different teams but establish a step-wise integration of components to secure the functionality dependence problems. Stabilization is improved by the technical environment of the components.

Category C type components allocated to different teams and must be integrated late at the end of all components completion because these components have no dependence. At the time of type c components integration, other components type A (already inter integrated) and type B integrated with each other for the complete shape of project.

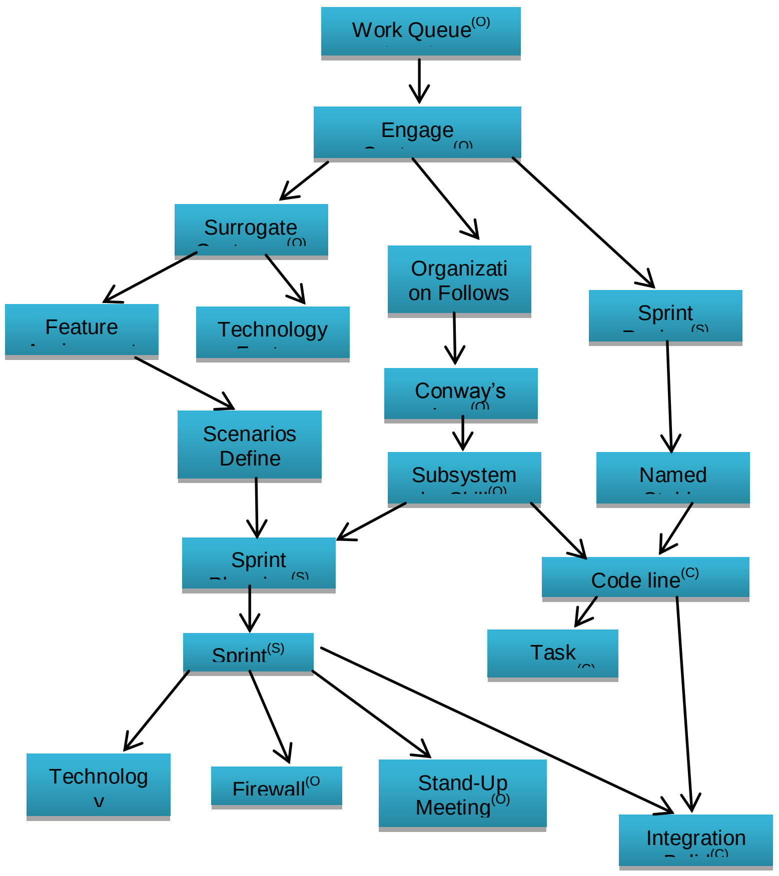

Figure 4: Proposed Pattern Language Implementing Technology Integration Pattern

3.3.4.Resulting Context

In a distributed environment, project is decomposed into subsystem and subsystem further into components. Every divided subsystem is allotted to a separate team. After decomposition there exists dependency between components related to technology. Kept an Integrator manager will help the development team to optimize and control the integration activities within the subsystem and with other subsystem. This effort will help to keep view the on-going integration activities. Such actions are not normally measured in its place unstated as regular factoring during single iteration. The suggestions for the performance measures and quality would be dedicated first to the whole system feature and then secondarily to the subsystem to reduce suboptimization. Usually benefit from this pattern is mainly on planning of characteristics during Sprint planning (typically Architect, Scrum Master and each team legislative body is included). After implementing the Technology Pattern the complete purposed language is given in Fig 4.

The more assured the project is in requirements and technology; it reduces the intermediate integration and parallel work management. However, for less predictable projects, iterative method is more suitable. This integration mechanism makes it possible to central improved on the whole effectiveness, decrease risk (e.g. technology incompatibility among subsystem) and stabilization of project.

IV. Results

We selected 20 people for participation but only 15 peoples give response back for contributes to the survey. So the reply ratio is 80%. We selected the 20 question for survey questionnaire and these questions were no inspirited from previous work done.

Survey questions mostly related to steps software decomposition and integration related to this new pattern. Also includes questions about the effect of this pattern on different phases of development. Mainly question related to the stability of the project in distributed scrum environment.

For this work data is collected from survey and face to face interview, from my work the first point which we got that technology factor for software project is almost ignored in comparison with functionality factor. It is also accepted by survey participants this ignorance of technology cause the unstable of the software. So there is a need to consider technology as a major factor during development especially for those projects which work on different technology (platform).

4.1.Technology Pattern

It is accepted by people during survey that the new purposed pattern (decomposition and integration) have created positive effect on overall development which is described in the following Fig 5**.**

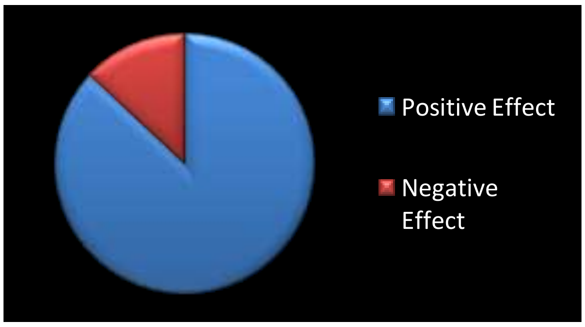

Figure 5: Effect of New Technology Pattern on Software Project

4.2.NewPattern Effect on Maintainability

In distributed scrum, teams are physically scattered, in such a situation the components development creates problem for teams if decomposition and integration of these components is not properly handled. The new purposed pattern provides a complete mechanism for such distributed team for decomposition and integration. It is cleared from the collected data this pattern helps teams to decrease issues related to maintainability of the project. This pattern provides a mechanism to categories the project components on the basis of their technology platform. This type of pre development mechanism reduces the integration issues which is the major problems for components and distributed development. This pattern increases the maintainability of project to 73% which is clarified in Fig 6. Maintainability is the major factor to ensure the stability of a project.

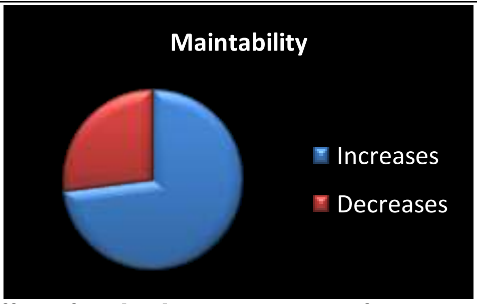

Figure 6: Effect of Technology Pattern on Software Maintainability

4.3.New Pattern and Stability of Distributed Projects

This pattern helps scrum distributed teams to adopt a mechanism of decomposition and integration of components. This pattern decreases the maintainability issues for physically scattered teams and increases the stability and overall life of a project. Survey peoples accepted that by this pattern, there 80% chances to increase the stability and life of a project which is given in Fig 7.

.

4.4.New Pattern Effect on Stability

Due to a complete mechanism for integration, this pattern helps to increase future scalability of project. It is cleared from collected data, there are 87% chance to support the project scalability by the implementation of this pattern which is given in Fig 8.

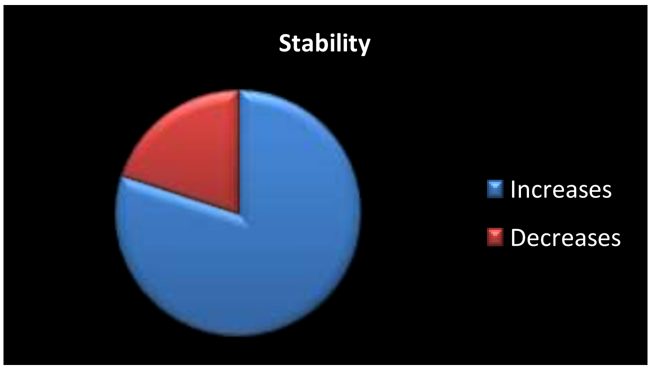

Figure 7: Effect of Technology Factor on Project Stability

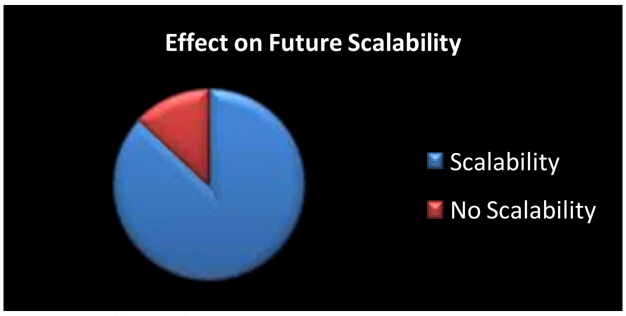

Figure 8: Effect of Technology Pattern on project Scalability

4.5.New Pattern Effect on Project Risk Management

With the attainability of this integration mechanism supports the project risk management. It provides a proper support to the project with the handling of integration issues from the start, so reduce the chances of risk to the project. The collected data shows, this pattern reduces the risk factor to the extent of 60% which is shown in Fig 9.

4.6.Adaptation of New Pattern

Adaptation of this pattern is depending on nature of project and teams. This pattern involves components integration so implementation on wrong project creates problems for teams. So there is a need to carefully adopt this pattern depending on the nature of project. From interviews and survey, major development peoples like this pattern and 73% people said this pattern is easy to adopt which is given in Fig 10.

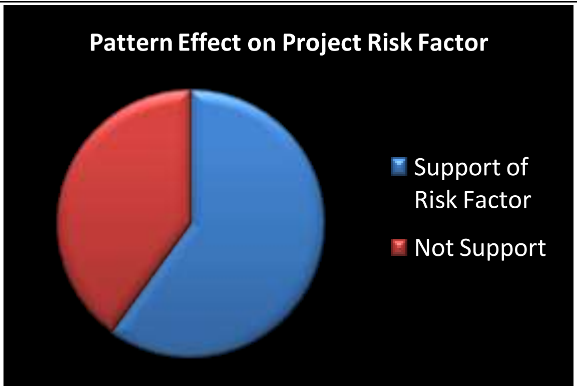

Figure 9: Pattern Effect on Risk Factor

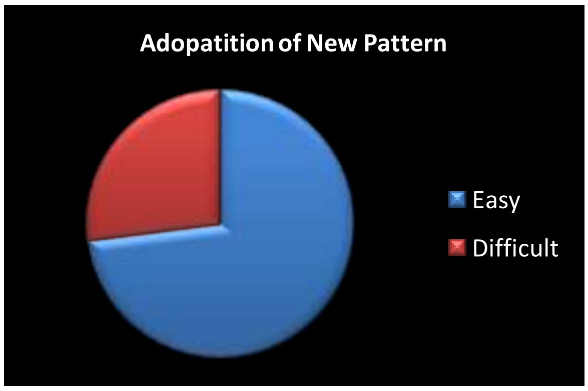

Figure 10: Adaptation Progress of New Project

V. Discussion

In this research we represented an application of the Scrum methodology, with Organizational patterns in distributed software development. This work described the use of three organizational patterns with scrum methodology patterns, with one being proposed as new patterns ("Technology Decomposition and Integration Pattern"). The proposed pattern ***Technology Decomposition and Integration Pattern*** is applied when development teams are separated and present technology with such distributed development. This pattern provided a way to decompose, prioritize and refine a story in context of technology and described a mechanism for integration of decomposed subsystems.

Survey implications show that this pattern has overall positive effect on a project. This provided a pattern for decomposition and integration of components. The major issues for distributed teams are the problems related to the integration when these teams work on components developments. This pattern makes development work easy for distributed teams, reduces the testing problems and increases the maintainability of project which in turn increase the stability and life of project. This pattern also has positive effect on future scalability of project due to decomposition and integration mechanism. The overall effect on project is also described in Fig 11.

This pattern has shown two important results. First, technology decomposition and integration pattern was recognized. The pattern and their demonstrations have shown that there are many advantages of this pattern. They also presented that in software engineering bases support for integration is variable for technical integration. Second, practical implications have shown that, integration arises neither easy nor free. This pattern helps to made software project more stable by proper integration of technology. In this pattern for **Type C** components integration, the internal development project risk increases due to early and step wise integration, but whole system implementation risk decreases. For **Type A** and **Type B** components integration, the internal development project risk decreases and whole system implementation risk increases and also greatly increases the complication of the development project. The decomposition and integration pattern also represented a logical method that helps the distributed scrum development team in the planning for iterative development projects. Selecting this pattern in scrum will help the practitioner to investigate the forces within it to handle some of the integration risks. However, carefully evaluate the application area for this pattern. Choosing the correct pattern wants compassion to context against project encapsulation.

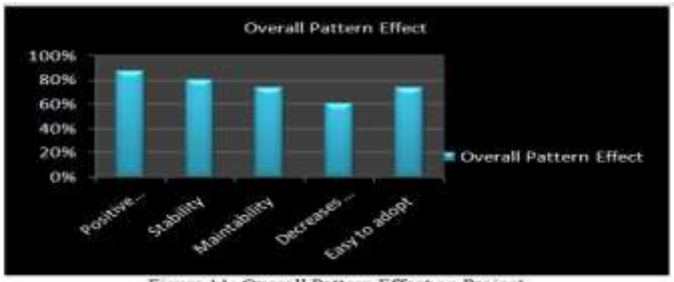

Figure 11: Overall Pattern Effect on Project

VI. Conclusions

Survey results show that this pattern has overall positive effect on a project. This provided a pattern for decomposition and integration of components. The major issues for distributed teams are the problems related to the integration when these teams work on components developments. This pattern makes development work easy for distributed teams, reduces the testing problems and increases the maintainability of project which in turn increase the stability and life of project. This pattern also has positive effect on future scalability of project due to decomposition and integration mechanism. The decomposition and integration pattern also represented a logical method that helps the distributed scrum development team in the planning for iterative development projects. Selecting this pattern in scrum will help the practitioner to investigate the forces within it to handle some of the integration risks. However, carefully evaluate the application area for this pattern. Choosing the correct pattern according to the nature of project will provides positive results otherwise it will create problems.

6.1. Future Work

Unfortunately I had no chance to experiment my findings on a development project, which would have been appreciated. However, most of the participants of survey and interview accepted this pattern the 'best practices' so this pattern is likely to be useful for distributed scrum development.

References

[1] Barton, A., R.A., Bartels, and M., Jenkins,2005. *Implementing software metrics in an agile organization*.Thesis (Master), University of Costa Rica.

[2] Blagojevi, M., Sullivan, K., andKazman, R., 2008. Identified four different models for applying scrum based on two parameters.*Third International Workshop*, January 28-30 2009, University of Sevilla, Spain.

[3] Beedle, M., M., Devos, Y., Sharon, K.,Schwaber, andJ., Sutherland, 2000. *An extension pattern language for hyperproductive software development.* Pattern languages of program design, Harrison, N., Foote, B. and Rohnert, H. ed. Boston: Addison-Wesley, Reading Mass, pp. 637-651.

[4] Begel, A. andN.,Nagappan, 2007. Usage and perceptions of agile software development in an industrial context: *In: Empirical software engineering and measurement*, First International Symposium, December 2007 London. Oxford, pp. 255-264.

Boehm, B., 2002. *Get ready for agile methods with care*.IEEE Computer, 35(1), 64-69.

[6] Cho, J., 2002. *An interactive computer and literacy instruction system*.Master Report, Utah University.

[7] Cho, J., 2008. Issues and Challenges of Agile Software Development with Scrum, 2008, Issues in Information Systems,Prentice Hall, 1X(2):350-368.

[8] Coplien, J.O., and D.C., Schmidt, 1995. Pattern Languages of Program Design (A Generative Development-Process Pattern Language), Addison and Wesley, Reading.

[9] Dent, A., 1996. "Solo Software Engineering", Software Development, CMP Technology, http://www.ddj.com/architect/184415530.

[10] Fraser, S., B., Kent, C., Bill, M., Tim, N., James, and P., Charlie, 2003."Test Driven Development (TDD)."In M. Marchesi and G. Succi, eds., XP 2003, LNCS 2675, pp. 459-462, 2003.© Springer-Verlag, Berlin and Heidelberg.

[11] Hossain, E., M.A., Babar, andH.P.,Young, 2009. *Using scrum in global software development: A systematic literature review*. IEEE Computer Society, 6(40), 175-184.

[12] Jakobsen, C.R.,andK.A., Johnson, 2008. *Mature agile with a twist of CMMI*. IEEE Computer Society, 13(1), 212-217.

[13] Potter, N.,andM., Sakry, 2003. *Implementing scrum (agile) and CMMI together.Guidelines for process integration and product improvement.*Boston: Addison-Wesley, 16(2), 102-106.

[14] Qumer, A.,andS., Henderson, 2006. Comparative evaluation of XP and scrum using the 4D Analytical Tool (A-DAT) .European and Mediterranean conference on Information System (EMCIS), July 6-7 2006, Costa Blanca, Alicante, Spain, 1-8.

[15] Schwaber, K., 1995. *Advanced development methods*. Available from: http://jeffsutherland.com/oopsla.

Striebeck, M., 2006. "Ssh! We're Adding a Process..." in Agile 2006 Minneapolis: IEEE.

[17] Sutherland, J., 2001. *Agile can scale: inventing and reinventing scrum in five companies*. Cutter IT Journal, 14(12), 5-11.

[18] Sutherland, J., 2004. "Agile Development: Lessons Learned from the First Scrum," Cutter Agile Project Management Advisory Service: Executive Update, vol. 5, pp. 1-4.

[19] Sutherland, J., 2005. Future of scrum: parallel piplining of sprints in complex projects. *In: Agile Conference*, 1-5 November, IEEE Computer Society.

[20] Sutherland, J., A., Viktorov, andJ.,Blount, 2006. Adaptive engineering of large software projects with distributed/outsourced teams. *In: International Conference on Complex System*, Boston, MA, USA.

[21] Sutherland, J., A., Viktorov, J., Blount, andN.,Puntikov, 2007. Distributed scrum: agile project management with outsourced development teams.*In: Hawaii International Conference on Software Systems*, 6-9 October, Big Island and Hawaii.

Sutherland, J., 2008.*Scrum and organizational patterns*. London, 75-79.

[23] Sutherland, J., S., Downey, andB.,Granvik, 2009. *Systematically achieving hyperproductivity agile*. Chicago.

[24] Sutherland, J., C.R., Jakobsen, and K.A., Johnson, 2010.Mature agile with twist of CMMI. *Agile 8th Conference 2008*, IEEE, 212-217.

[25] Yaggahavita, H.D., 2011. *Challenges in appling scrum methology on culturally distributed teams.*Thesis (Master), Sheffield Hallam University (SHU).