ABSTRACT:
Self-organizing teams are at the spirit of agile software development. Self-organizing Agile teams are composed of individuals supervise their own workload shift work among themselves based on need and best fit and participate in team decision making. Self-organizing teams must have frequent focus, mutual trust, respect and the capability to systematize frequently to meet new challenges. The scrum method particularly mentions self-organizing agile teams and the concept of empowered teams has recently been added to XP. Self-organizing teams are not only seen as allowing Agile engineering practices but also as capturing the spirit of Agile values and principles which focus on human and social aspects of software engineering. Self-organizing teams is one of the principles behind the agile proposal and have been recognized as one of the significant success factors of agile projects. This paper presents the results of a qualitative Grounded Theory (GT) study involving agile practitioners from different software organizations in New Zealand and India conducted over a period of four years. As a part of the study semi-structured, face-to-face interviews were conducted with Agile practitioners in the New Zealand and Indian software industries using open-ended questions.

KEYWORDS: Self-organizing, team roles, software engineering, Agile software development, grounded theory.

INTRODUCTION:
Self-organizing teams have taken centre stage in software engineering when they were included as a hallmark of agile methods. Despite the extended and wealthy history of self-organizing teams and their recent popularity with agile methods there has been little research on the topic within software engineering. Mainly there is a lack of research on how agile teams systematize themselves in practice. Through a Grounded Theory research involving agile practitioners from software organizations in New Zealand and India over a period of four years we recognized informal, implicit, transient and spontaneous roles that make agile teams self-organizing. These roles Mentor, Coordinator, Translator, Champion, Promoter and Terminator are focused toward providing initial leadership and encouraging continued devotion to Agile methods effectively managing customer expectations and coordinating customer collaboration protected and satisfying senior management support and identifying and removing team members threatening the self-organizing ability of the team. Understanding these roles will help software development teams and their managers better understand and carry out their roles and responsibilities as a self-organizing team. Self-organizing teams have been documented and studied in various forms as independent groups in socio-technical systems, enablers of organizational theories, agents of knowledge management and as examples of complex-adaptive systems. In particular while there an increasing interest in research on agile software development teams in common there is a dearth of research on the specific topic of self-organization in Agile teams and how Agile teams organize themselves in practice.

RELATED WORK:
A broad number of researchers have walk around team roles and dynamics. Belbin suggests nine team roles based on behaviour plant, resource investigator, coordinator, shaper, monitor evaluator, team worker, implementer, completer finisher and specialist. A coordinator in Belbin’s team roles theory focuses on team’s objectives and delegates work. The Coordinator role recognized in our research on the other hand helps coordinate between the team, their customers and does not delegate work. A key training of self-organizing agile teams is self-assignment. Specialist in Belbin’s theory focuses on a particular area of know-how and has a propensity to value their specialization over team goals. In self-organizing Agile teams nevertheless team members balance between cross functionality and specialization while remaining committed to the team goal. Five boundary-spanning roles have been identified as means to give confidence communication across boundaries ambassador, scout, guard, sentry and coordinator. Some studies have described individuals supporting customers by translating technical language to business language. In
dissimilarity our Translator role was able to achieve two way communications between the development team and their customers by translating business language into technical language and vice versa.

EXISTING METHOD:
The obstinacy of the GT method is personified by the steady comparison method. This procedure is repeated every time a new category is establish or there are changes in an existing category or new properties of an existing category is discovered leading the researcher to return previously coded transcripts to see if they have the new property. The explanation were analyzed and compared to the concepts derived from the interviews. When further data collection and analysis on a particular category leads to a point of deteriorating results the category is said to have reached academic saturation. The researcher can discontinue collecting data and coding for that category.

DISADVANTAGES:
The comments did not disagree with but rather supported the data provided in interviews thus reinforcement the interview data. In this investigate the last few interviews provided no new nearby into the existing categories which were obvious indication of theoretical saturation.

PROPOSED METHOD:
Agile methods were developed as an answer to the weaknesses of traditional software development models. Agile methods perk up over the traditional software development models by helpful changes through an iterative and incremental style of development allowing every iteration to focus on a small set of functionalities prioritized by the customer. Agile methods support continuous customer involvement and feedback and permit the customer to prioritize the features they want developed first. Some flavours of agile methods include Dynamic Software Development Method (DSDM) referred to as the first agile method. Crystal a family of methodologies consisting of a number of methods and principles for customizing them for particular projects. Feature Driven Development (FDD) which focuses on features-based division of work and Adaptive Software Development (ASD) which focuses on concepts and culture and creating emergent order out of confusion.

ADVANTAGES:
The main beliefs behind the Agile Manifesto comprise fast, frequent, consistent and continuous liberation of working software, responding to changing requirements, encouraging effective communication, motivated and well supported self-organizing teams.

SELF ORGANISING TEAM ROLES:
The self organizing Agile team roles Mentor, Coordinator, Translator, Champion, Promoter and Terminator are focused specially toward self-organization and are execute over and above the organizational roles. The self-organizing roles are relaxed and implicit because unlike organizational roles they are not properly designated to the individuals who play them. The self-organizing roles are temporary because unlike organizational roles they emerge in response to challenges faced by the agile team and vanish or become dormant as the problems subside. The self-organizing team roles are spontaneous because unlike organizational roles they are instinctively picked up by different members of the team.

PROVIDING INITIAL GUIDANCE AND SUPPORT:
Most team members observe the Agile practices to be simple sufficient to comprehend but when it comes to implementing them on a daily basis they need guidance and support. The Mentor oversees the new team as they begin to practice agile software development on a day to day basis. The Mentor makes known the team with the Agile Manifesto values and principles and informs them of one or more particular agile methods such as Scrum and XP. The theoretical knowledge of agile software growth and the practices of particular agile methods are imparted by the Mentor in several ways. Some Mentors have familiar talks with their teams while others conduct more formal training sessions spanning a few days.

TEAM CONFIDENT:
In relatively new teams usually less than a year of experience the role of the Mentor is engaged up by experienced agile coaches who exhibit a firm understanding of both agile methods and their teams’ problems. These Agile coaches are often employed on a contractual basis to direct the new team during the initial stages of practicing agile software development. In more mature agile teams fluent in the use of Agile practices for generally more than a year though the role of the Mentor is taken up by anyone in the team with broad experience in agile software development. For instance in one of the Indian Agile organizations most members have numerous years of experience in agile software development and do not need a full-time Mentor.

COORDINATOR:
Agile methods get bigger the customer role within the complete development process by involving them in writing user stories, discussing product features, prioritizing the feature lists and providing quick feedback to the development team on a regular basis. These combined activities are hard to organize with the customer for various reasons such as physical distance between the development team and their customers, lack of time commitment on the part of the customers and ineffective customer representation. The Coordinator roles appear on agile teams to conquer these challenges and make easy collaboration with customers.

IMPLICATIONS FOR PRACTICE:
One of the characteristics of self-organizing teams is their ability to react spontaneously in response to challenges. In an Agile environment teams can anticipate to get involved in a lot more practices than just coding and testing. These practices include group programming as compared to working in isolation, daily stand-ups meetings and the use of information radiators to promote clearness, collective decision making and self-assignment. In the nonexistence of a manager that handles external relations for the team members should be prepared to take on the interfacing roles of Coordinator and Translator. At first individuals with good communication skills will find themselves taking on these roles. Likewise team members should be ready to champion their teams with senior management or encourage their teams with customers as required by playing Champion and Promoter roles respectively. Over time all members can expect to take up any or all of these team roles as needed.

RESEARCH METHODOLOGY:
Researchers employ qualitative research methods to develop an in-depth understanding of the phenomenon being explored. Due to little existing academic theory available to explain the issues faced by practitioners of distributed agile projects, we have decided to use a systematic qualitative research method called Grounded Theory to investigate our research. Moreover, lately Grounded Theory is being increasingly used to explore the social nature of software development of agile teams. Grounded Theory allows the theory, which is grounded in data systematically gathered and analysed, to emerge naturally. The emergent theory richly describes the agile practitioner practices and allows them to further understand their situation in managing distributed projects. Grounded Theory does not aim to the reveal accurate description but rather to conceptualise the transcending abstraction regardless of the time, place and people. The first stage in our research is the selection of research question. Although Grounded Theory does not encourage preliminary literature review and predetermined research problem yet a basic phenomenon of interest need to be recognized. The second stage in our research is data gathering. Though the fundamental notion of Grounded Theory is “all is data”, our primary source of data for the research is the semi-structured interviews with agile practitioners. The third stage is data coding which is a three-step process: (a) open-coding yields many concepts and compels there searcher to categorize and describe the phenomenon, (b) axial-coding relates the codes and describes the relationship between concepts, and (c) selective-coding uses the most frequently appearing open-codes to sort, synthesize and conceptualise the large amount of data extracted from the interviews. The final stage is validation. Validity is determined by fit, relevance, work ability and modifiability of the emergent theory.
confidentially until completion of research. Soon thereafter, these materials will be destroyed promptly.

The Emergence of Theory
Grounded Theory, being a perspective based methodology, requires constant comparison of data being collected, coded and analyzed than simple inspection of data to discover the latent theory of the phenomenon being researched. Therefore, we will continue to conduct more interviews and analyze them until the emergent theory fits the agile practitioners.

6. PROGRESS TO DATE:
We have conducted semi-structured face-to-face interviews using open ended questions with eligible practitioners from New Zealand, USA, UK, Argentina, Brazil, Germany. The interviews were all pre-scheduled and voice recorded where permission was granted. The participants were Scrum Masters, Agile Coaches and Senior Developers with many years of hands-on experience in working with globally distributed teams using combinations of XP and Scrum. We have transcribed all the interviews and analyzed those data using principles of grounded theory. The open–coding yields many concepts that enabled us to categorize and describe the nature of distributed agile projects. By performing selective-coding, we have identified categories. We have reported that trust is one of the fundamental factors that determine the success or failure of distributed agile projects, and presented several strategies employed by agile practitioners in successful distributed projects. We plan to interview more agile practitioners who are involved in distributed agile projects. We also plan to conduct several follow-up interviews with the existing participants. We will adapt our future interview questions to focus on the emerging categories.

LIMITATIONS TO VALIDATE:
The extremely elevated numbers of variables that affect a real software engineering project make it hard to recognize the collision that any one factor has on the success or failure of the project. The self organizational roles, practices and factors authority self organizing Agile teams however were obviously evident. A Grounded Theory research study produces a midrange theory which means that while the theory is not claimed to be generally applicable it can be customized by constant comparison to accommodate more data from new contexts. A key contribution of a GT study carried out correctly is that it focuses on conceptualization and produces flexible modifiable concepts with immense grab.

FUTURE WORK:
In Agile organizations where all software development is done by numerous self-organizing agile teams the self organizational roles at the team level need organization wide complements at the organizational level. Since all teams are self-organizing the need for mentoring, training, securing and coordinating customer’s collaboration and removing cultural oddity becomes organization-wide concerns. In reply the self-organizational team roles of Mentor, Coordinator, Translator, Champion, Promoter and Terminator can be reflected at the organizational level. The presences of these organization-wide roles were indicated in two mature agile organizations toward the end of this research. Future work could study agile software development companies to discover such organization-wide roles that enable self organization at an organizational level.

CONCLUSION:
A thorough re-evaluates of self-organizing teams from multiple perceptions and discussed the implications of our findings for practitioners. Understanding these roles will help software development teams and their managers better understand and implement their roles and responsibilities as a self-organizing team. Self-organizing teams are at the heart of agile software development. The lack of research into self-organizing teams in software engineering means there is little experiential confirmation to explain how agile teams put in order themselves in practice. Through a Grounded Theory study involving Agile practitioners from software organizations in New Zealand and India conducted over a period of four years we identified six informal, implicit, transient and impulsive roles on Agile teams that make them self organizing.

REFERENCES:


Mr. K. Raja Sravan Kumar, well known Author and excellent teacher received B.TECH and M.Tech (SE) from JNTUK university is working as Assistant Professor and, Department of IT. M.Tech Software engineering, Godavari Institute of Engineering and Technology. He is an active member of ISTE. He has 5 years of teaching experience in various engineering colleges. To his credit couple of publications both national and international conferences/journals. His area of Interest includes Software Engineering, Software process and project Management, Flavors of Cloud Computing and Web technology and other advances in computer Applications.

Ms. K. Rama Devi is a student of ADITYA Engineering College Surampalem, Kakinada. Presently she is pursuing her M.Tech [Software Engineering] from this college and she received his B.Tech from Nagarjuna Institute of Technology and Sciences, affiliated to JNT University, hyderabad in the year 2008. Her area of interest includes software engineering and Object oriented Programming languages, all current trends and techniques in Computer Science.