Master Thesis
Software Engineering

September 2012

Information Visualization for Agile Development in Large-Scale Organizations

Numan Manzoor and Umar Shahzad

School of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona
Sweden.

This thesis is submitted to the School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering. The thesis is equivalent to 2 x 20 weeks of full time studies.

Contact Information:
Authors:
Numan Manzoor
Numan.manzoor@hotmail.com

Umar Shahzad
Umar_sh@hotmail.com

University advisor:
Dr. Samuel A. Fricker
School of Computing (COM), BTH
samuel.fricker@bth.se

Internet : www.bth.se/com
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

ABSTRACT

Context: Agile/lean development has been successful situations where small teams collaborate over long periods of time with project stakeholders. Unclear is how such teams plan and coordinate their work in such situations where interdependencies with other projects exist. In large organizations, scattered teams and complex team structure makes it difficult for every stakeholder to have a clear understanding of project information. These factors make it difficult for large-scale organizations to adopt the agile/lean development paradigm.

Objectives: The goal of conducting this study is to find the information visualization techniques that ease or resolve the challenges of agile development in large-scale organizations. The study reports the challenges of agile development and information visualization techniques in literature and reported by industrial experts. Additionally, proposed a guideline that how information visualization technique can be used to ease or resolve related challenge of agile development.

Methods: For this particular study, two research methodologies are used; Systematic Literature Review (SLR) and Industrial Survey. Two SLRs are performed for finding 1) challenges of agile development and 2) information visualization techniques in agile development. Data sources like Engineering Village (Inspec/ Compendex), IEEE Explore digital library, ACM digital library, Science Direct, ISI-Web of knowledge; Scopus were used to select primary study. Industrial survey was conducted in order to obtain empirical evidence to our findings. In survey, mainly questions were related to challenges of agile development and information visualization techniques practiced by industrial experts.

Results: 84 different challenges of agile development found in literature and by applying grounded theory we found 9 distinct categories of challenges. There were 55 challenges reported by industrial experts in survey which later grouped into 10 distinct challenges. 45 information visualization techniques found in literature and grouped into 21 distinct technologies. There were 47 different information visualization techniques reported by industrial experts. When we grouped these techniques there were 9 distinct technologies found by applying open, axial and selective coding of grounded theory

Conclusions: Systematic Literature Review and Industrial Survey confirmed that information visualization techniques can be used to ease or resolve challenges of agile development. Along with other visualization techniques, Data Flow Diagrams, UML, Use Case Diagrams, Burn Down Charts, Scrum Story Board, Kanban Boards and Gantt Chart are highly reported techniques found through systematic literature review and later confirmed by industrial experts. On the other hand, through survey we found that industrial experts mainly rely on informal and customized information visualization techniques to visualize information.

Keywords: information visualization techniques, agile development, challenges of agile

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Agile software development is a group of software development methodologies that are based on iterative and incremental development processes. In agile software development requirements and solutions evolve through collaboration between self-organizing and cross-functional teams [1]. Dealing with agile methodology under best possible conditions, can produce results in shape of high quality and efficient product [2]. Practitioners are more focused on people rather than on reporting deliverables are often working fine and at the end produce successful working products. A number of software development organizations have started to deploy agile methodologies as a main development practice to develop software products [3]. Agile methodologies are considered as development methodology to avoid those overheads which are imposed by traditional software development environment [4].

Along with the other traditional plan driven development methods i.e. Water fall model [5], V-Model [6] and Rational Unified Process (RUP) [7], induction of different agile methods like Scrum [8], Dynamic System Development Method (DSDM) [9], Extreme Programming (XP) [10], Crystal Methodologies, Feature-driven Development [11] and Lean [12], significantly change the concept of software development . These methods have drawn the attention of practitioners in software industry by providing flexibility to welcome late requirements, effectively dealing with budget constraints and promising results in product delivery [13] [14] [15]. Agile methods were developed by a group of industrial experts and researchers in shape of “Agile Manifesto” [16]. The main differences between agile and plan driven development are in shape of management style, fundamental assumptions in product development, communication, knowledge management, quality control and organizational structure [17][18]. We found the use of agile methods in different domains, like in large and distributed environments, embedded and telecom systems [19]. By going through current literature, evidence shows that agile methods are not only suitable for small team structure but now also gear attention to effectively apply on large and complex applications [8].

Information visualization deals with visual representation of abstract data/ information to support better understanding of data/ information [20]. The abstract data/ information can consist of both numerical and non-numerical data. In software development, information visualization techniques help stakeholders to get a bird eye view of the developing system during in development and facilitate them to make decisions to align with business goals. Information visualization not only provides visualized form of information/requirements but also facilitates project members to manage their information [21].

1.1 Related Work

Agile/lean development has been successful in situations where teams collaborate over long periods of time with project stakeholders. The current agile methods are originally designed for small teams [22], while there are lots of challenges for the application of agile software development methods in large organizations [23]. In large scale organizations, by dealing with agile methodology is not an easy task to do where organizations have large and complex team structure, diverse backgrounds of team members, constant change in requirements, stakeholders synchronization and rapid development by keeping in view time-to-market aspect [24] [25] [26]. There would be a need to scale agile methods and build understanding of how success factors such as the alignment of the evolving software solution with business goals and stakeholder expectations ,can be monitored and managed by applying information visualization techniques [27].

There are different information/requirements visualization techniques [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] used in agile development. These techniques are applied for business modeling, project tracking and work status. In current literature studies discussed information visualization techniques either has weak empirical evidence or discussing single technique. These studies did not able to clearly describe how these visualization techniques can be used to ease or solve the challenges of agile development in large-scale organizations [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38]. There is a need to explore these visualization techniques across their functionalities and application in agile development, and how these techniques ease or solve the challenges of agile development in large scale organizations. The motivation behind the use of Information/requirement visualization techniques is that, these techniques provides a realistic picture of functional requirements which is difficult to extract from the bulk of documentation and instantly provide a clear view of ongoing processes to the project stakeholders. This practice enables them to take development decisions.

Benefit and pitfalls of Story board are briefly discussed in the context of Extreme Programming (XP) for project tracking and work status [29]. Author has produced the user story tool named as “DotStories”, based on requirements after investigating the team working behavior on remote locations. Developed tool effectively simulate the process of iteration, planning and provide flexibility to record bulk of stories, and finally re-ordering or grouping development tasks in iteration planning.

In this paper [28], a case study was conducted in industrial setting to closely monitor the different challenges faced by Scrum teams in multi project or multi team behavior. They experienced that Burn down charts are supposed to be quite helpful to track work progress in different sprints, adjust or meet the sprint commitments and to build team coordination efforts. Results have shown some negative behavior in careful tracking because team members sometimes get bored by keep playing with childish cards and tracking of work progress.

In large scale organizations, distributed team structure has always put constraint in binding of different teams and keep intact their coordination and communication throughout the project. N.Naslia Khairuddin et al. [38] worked on requirements visualization techniques to transform the verbal and written requirements into some visual presentation form. They used three different visualization techniques e.g. 3D visualization technique, Behavioral requirements specification approach and UML models, later on authors did comparative analysis based on their four selected attributes comprehensiveness, validation reports, ease of use, and ease of conversion to implement codes.

Authors conducted a case study to get brief insights into different visualization tools and techniques used by agile development in distributed development settings [33]. Along with other techniques, Burn-down charts shown significant results in the form of team progress in any iteration of the project. Study shows that burn-down charts are giving better overview of budget vs. time spent on iteration and help development team to budget reasonable time for the features.

Combining and applying different domains of hypermedia to support daily Scrum meetings discussed in [34]. Rubart et al. introduced change structure in Scrum to clearly model change in flexible way. Authors planned to make a cooperative task board for distributed Scrum development teams to register themselves on board and use interface for their Scrum meetings to give strength to Scrum sprint planning and improved Scrum retrospectives.

A process is discussed for functional requirements visualization in terms of behavioral interactions, in business modeling using Unified Modeling Language (UML) model [39]. The motivation behind their research was that Model driven development (MDD) and UML have become increasingly common practice in the industry and software developers faced serious challenges to understand how UML model recognize system requirements. Authors proposed an alternative modeling approach named Requirements visualization of UML (REVU), a process helps users to create and visualize the witness scenarios in natural language when little information is available of the system and finally present scenarios in visual form. Here, witness means, a system executing a particular sequence of steps to fulfill a given functional requirements.

In paper [40], author conducted a case study to present the need of agile development methodology of product management in Cyber forensic (process of acquisition, seizure and analysis of digital evidence) tool designing. They observed that Storyboard can be effectively used to improve the visibility of the development progress of the project. This sufficiently improves the team's daily productivity by constantly reviewing the sprints features progress and by giving work accomplishment in shape of demos at the end of each sprint.

Author discussed in [41], agile transformation at Borland development organization as a part of effort to minimize development cost, boost in quality and efficiency, and present operational inaccuracy. At the end, company reaped benefits in shape of product quality, productivity and most importantly in team's morale. The study results shows that agile transformation in large scale organization is not a easy task to do and need long term ongoing process to implement fully functional agile development processes, company-wide willingness to adopt change in defined organizations rules/standers and to overcome team culture differences.

In [42], requirements specifications traceability with respect to agile development in large scale organizations is discussed that enables change impact on the requirements, rapid assessment, and alive customer's interest throughout the project. They used tool based approach (ECHO) for scalability of agile requirements gathering practice.

In this paper [43], author proposed a classification and some preliminary principles for designing and implementing visual representations of formal specifications. These principles are proven with examples of visualizations they created, while trying to understand a formal specification of the MD-11 Flight Management System. Author concluded that the use of interactive, computer-supported, visual representations of the specifications that will help requirements engineers to read, create and understand formal specifications. Martin S. Feather et al. [44] also used a risk based decision method for development. They supported their process with the help of customized visualization software. In requirements gathering, visualization was performed to scrutinize the requirements status to get a clear picture for decision making on gathered requirements.

1.2 Research questions / Hypotheses

RQ1: What are the challenges for agile development in large-scale organizations?

- **RQ1.1**: What are the challenges for agile development in large-scale organizations available in literature?
- **RQ1.2:** What are the challenges for agile development in large-scale organizations currently faced by industry?

RQ2: Which are relevant information visualization techniques that can be used for agile development in large-scale organizations?

- **RQ2.1:** Which are relevant information visualization techniques available in literature?
- **RQ2.2:** Which are relevant information visualization techniques currently practiced by industry?

In RQ2 by “relevant” we mean, those information visualization techniques that can be used, to effectively deal with the challenges identified in RQ1, like Burn down charts help to visualize the estimation process and reduce the communication challenge of agile in large scale organizations [38].

RQ3: How can identified information visualization techniques ease or solve the agile challenges identified in RQ1?

The main motivation of these questions is to review current literature and by consulting industry experts to find the information visualization techniques that ease or resolve the challenges of agile development in large-scale organizations. By answering the above questions we would be able to find out that how information visualization will be useful to resolve challenges for agile development in large-scale organizations.

1.3 Aims and Objectives

The goal of this study is to find the information visualization techniques that ease or resolve the challenges of agile development in large-scale organizations.

- To achieve above goal, we find out the important challenges for agile development in large scale organizations, through literature review and by consulting industrial experts.
- Similarly, we find out relevant information visualization techniques for agile development through literature review and by consulting industrial experts.
- Finally, we develop a guideline, to explain that which information visualization technique can be used, to ease or solve the identified challenges, of agile development.

1.4 Thesis Outline

Chapter 1, INTRODUCTION, provides an introduction to the area of this study.

Chapter 2, RESEARCH METHODOLOGY, describes the research design by describing the systematic literature review, industrial survey and grounded theory.

Chapter 3, RESULTS AND ANALYSIS OF SYSTEMATIC REVIEW, presents the results and short analysis of SLR for RQ1.1 and RQ2.1.

Chapter 4, RESULTS AND ANALYSIS OF INDUSTRIAL SURVEY, presents the industrial survey results.

Chapter 5, ANALYSIS, provides the comparison of literature review and survey results and also validity threats to this study.

Chapter 6, CONCLUSION, presents the discussion, mapping of research questions and future work.

1.5 Thesis Terminology

Terms/Abbreviations	Definition
GSD	Global Software Development
Agile Practices	Scrum, Extreme programming, Feature driven development, lean software development,
3C’s	Communication, Coordination and Control
Traditional industry practices	Waterfall Model, Iterative and incremental development, Spiral Model
XP	Extreme programming
SLR	Systematic Literature Review
RE	Requirement Engineering

Table 1: Thesis Terminologies

2 RESEARCH METHODOLOGY

A mixed approach, qualitative and quantitative, is used in this study. The systematic literature review, expert's survey and industrial case study are the main activities to answer the research questions of this study. This chapter describes the research design for this study.

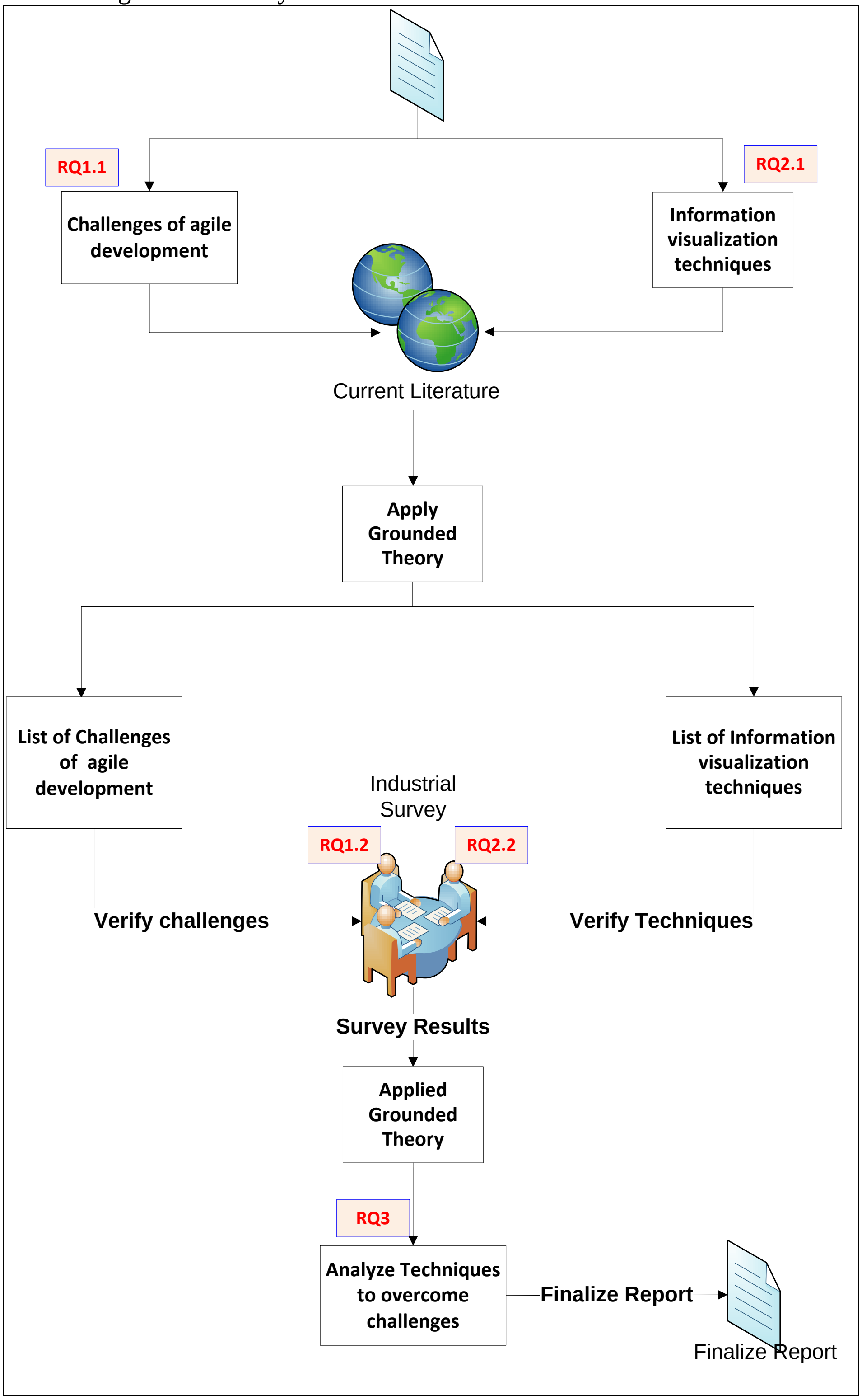


Figure 1: Flow of the study

2.1 Systematic Literature Review

Systematic Literature Review (SLR) is defined in [45] as "identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest". In systematic literature review, researchers identify, evaluate and interpret all available research that is relevant to a particular research area [45]. By using systematic literature review, researcher become able to define the explicit and rigorous criteria to identify, critically evaluate and synthesize available research literature [46]. Kitchenham et al. mentioned in [47], systematic literature review can be conducted to summarize existing evidences, identify any research gaps in existing research literature and later on propose a guidelines.

Motivation: Our motivation for selecting SLR as research methodology for answering RQ1.1 and RQ2.1 is that by using SLR, we can able to define explicit and rigorous criteria to identify and evaluate existing research literature. For this particular research study we used an authentic method to identify, analyze and interpret research literature. Systematic Literature Review according to guidelines of Kitchenham et al. allows us to define explicit and rigorous criteria to identify and evaluate existing research literature [47]

Alternatives: There are other types of literature review exits that can be used as an alternative of Systematic Literature Review: systematic mapping studies, narrative literature review and tertiary reviews. Systematic mapping study can be conducted where research area is very extensive and quite probability that there will be very less evidence available in literature. Narrative literature review summarizes the body of knowledge and draw conclusion about the research area [46]. Tertiary reviews can be used when there are large number of systematic literature reviews about the research topic already exists and analyzing their results and present conclusion [46].

Dismissal of Alternatives: For finding challenges of agile development and information visualization techniques available in current literature, we have to use an authentic method that defines rigorous criteria to identify and evaluate existing research literature. SLR helps us to critically evaluate and synthesize the research literature, that's why we selected SLR to review the literature. Conducting SLR is a repeatable process so that any researcher can follow these steps to get same research results [47].

2.1.1 Summary of existing SLRs for RQ1.1

In 2009, Peterson & Wohlin [48] piloted a case study to identify issues and advantages by using agile and incremental development in large scale environment. This study agreed on number of challenges found in available literature and only few challenges that were reported in case study are listed in current literature. Some identified challenges by implementing agile and increment development practices in large-scale environment are in shape of complex decision processes that in result per long requirement engineering process, hard to create requirement priority lists, long waiting time in development processes like design keep waiting requirements, lack of independent testing, packaging efforts increase and increase in configuration management efforts. Furthermore, authors also experienced that number of identified challenges had not been clearly addressed when scaling agile in large development environment which is more complex in nature. In addition to these issues, case study also mentioned some advantages of agile development in shape of better knowledge transfer, more precise requirements, reduce documentation by direct team communication, frequent deliveries gives early feedback, low volatility in requirements, low wastage in terms of discarded requirements due to enhanced requirement engineering process.

In 2008, Dyba & Dingsoyr [49] conducted a systematic review on agile software development that identified empirical studies up to 2005 and end up with 36 relevant empirical studies. The review investigates currently known benefits and limitation, and strength of evidence for agile methods. The research shows that there is a need to increase quality of empirical studies and form a common research agenda in area of agile software development.

In 2009, Hossain et al. [50] carried out systematic literature review on using Scrum in Global distributed development (GSD). They span their study to 366 papers and end with 20 papers that were more relevant to their research area. Authors study highlighted the challenges faced by using Scrum in GSD environment where teams dispersed in nature and challenging task is to maintain communication, coordination and control in the development cycle. The motivation behind the chosen agile methodology was that Scrum has focused on day to day project process management. Additionally, the best practices deals with known challenges have been extracted. The presented conclusion and guidelines can guide both researchers and practitioners to use and understanding of Scrum practices in GSD setting. Moreover, project managers can also benefits from the synchronized knowledge about identified strategies to deal with various challenges by using Scrum in GSD.

2.1.2 Review Protocol

Review Protocol describes the detailed plan for conducting SLR and provides a process/method to select primary studies which can further reduce publication biasness [45].

2.1.2.1 Objective

The purpose and objective of the Systematic Literature Review is to gather and summarize the available literature related to challenges of agile in large scale organizations and information visualization techniques. For this particular study, there is a need to perform two SLRs: one is for identifications of challenges of agile in large scale an organizations while second is to find out information visualization techniques available in current literature.

2.1.2.2 Search Strings for RQ1.1

For searching the literature, the process starts with development of search strings. The following steps are performed in order to develop the search strings. These steps are performed according to the guideline of Biolchini et al.[51].

1) Identified the important search terms from research questions as keywords.
2) Used alternative words and synonyms for each keyword, in order to minimize the effect of difference in terminologies.
3) Organized keywords and synonyms with Boolean operators to form a search string.
4) Boolean “OR” is used for joining alternative keywords and synonyms.
5) Boolean “AND” is used for joining the keywords into search strings.

Research Question	Keywords from RQ	Synonyms or Relevant of keywords
What are challenges for agile development in large-scale organizations available in literature?	Challenges	Problems, Issues, solution, resolution
	Agile Development	agile, "extreme programming", xp, Scrum, "dynamic software development method", lean, "agile modeling", "iterative development", "iteration software development", "incremental software development"
	Large-scale organizations	"large-scale", "large scale", big, large, organization*, compan*, firms, industr*, "software companies"

Table 2: Synonyms or Relevant of keywords for RQ1.1

Search string for searching the databases were developed by concatenating the keywords, synonyms, relevant keywords with the help of Boolean operator “OR”. Because all identified keywords should be present in search string the Boolean operator AND was used. After concatenating the keywords the search strings look like:

((challenges OR problems OR issues) OR (solutions OR resolutions))
AND
(agile OR "extreme programming" OR xp OR Scrum OR "dynamic software development method" OR lean OR "agile modeling" OR "iterative development" OR "iteration software development" OR "incremental software development")
AND
("large-scale" OR "large scale" OR big OR large)
AND
(organization OR compan* OR firms OR industr* OR "software companies")*

Search Strings for SLR of RQ1.1

The search strings that were used for searching the existing SLRs or Systematic Literature Reviews in the current research literature are given below.

((challenges OR problems OR issues) OR (solutions OR resolutions))
AND
(agile OR "extreme programming" OR xp OR scrum OR "dynamic software development method" OR lean OR "agile modeling" OR "iterative development" OR "iteration software development" OR "incremental software development")
AND
("large-scale" OR "large scale" OR big OR large)
AND
(organization OR compan* OR firms OR industr* OR "software companies")*
AND
("Systematic literature review " OR "SLR" OR "literature review" OR "state of art" OR "state of knowledge")

Search Strings for RQ2.1

Research Question	Keywords from RQ	Synonyms or Relevant of keywords
Which are the most relevant information visualization techniques available in literature?	Relevant (Agile)	agile, "extreme programming", Xp, Scrum, "dynamic software development method", lean, "agile modeling", "iterative development", "iteration software development", "incremental software development"
	Information Visualization	Information/requirements Visualization
	Techniques	technique*, practice*, method*, tools, process, procedure

Table 3: Synonyms or Relevant of keywords for RQ 2.1

Search string for searching the databases were developed by concatenating the keywords, synonyms, relevant keywords with the help of Boolean operator “OR”.

Because all identified keywords should be present in search string the Boolean operator AND was used. After concatenating the keywords the search strings look like:

(Information OR requirements AND (agile OR "extreme programming" OR xp OR scrum OR "dynamic software development method" OR lean OR "agile modeling" OR "iterative development" OR "iteration software development" OR "incremental software development"))
AND
(visualization) AND (technique* OR practice* OR method* OR tools OR process OR procedure)*

Search Strings for SLR of RQ2.1
(Information OR requirements AND (agile OR "extreme programming" OR xp OR scrum OR "dynamic software development method" OR lean OR "agile modeling" OR "iterative development" OR "iteration software development" OR "incremental software development"))
AND
(visualization) AND (technique* OR practice* OR method* OR tools OR process OR procedure)*
AND
("Systematic literature review" OR "SLR" OR "literature review" OR "state of art" OR "state of knowledge")

2.1.3 Data Sources

Research articles related to software engineering can be searched in six different search databases. These databases are also recommended by Brereton et al.[52]. Following electronic databases are used to perform SLR to address our research questions.

- Engineering Village (Inspec/ Compendex)
- IEEE Explore digital library
- ACM digital library
- Science Direct
- ISI-Web of knowledge
- Scopus

2.1.4 Study Selection Criteria

The Study inclusion and exclusion criteria classify those primary studies that provide direct influence or evidence on the research questions [52]. Here direct evidence means, articles that have actual relevance within the context of research questions. To reduce the factor of likelihood biasness, study selection criteria should be well stated in the search protocol definition [52]. After applying search strategy on different search databases, we identified 908 papers to move further to apply inclusion /exclusion criteria. In the first step, we removed duplicates from the pool of identified papers and then evaluate on the bases of full text availability. Although, the search strategy was formed carefully to minimize out of scope and irrelevant papers in search results, around 85 papers were removed after applying title and keyword filter. Furthermore, we scanned each paper on the bases of title, abstract and conclusion, and at the end removed irrelevant research results. Finally, we had 62 papers ready for throughout study and in result 26 papers were selected for primary study. To ensure the search reliability and correctness of search results, following inclusion and exclusion process is used to extract our primary study;

2.1.4.1 Inclusion Criteria

- The selection of articles from publication year 2000 to 2011.
- Articles are available in full text.
- Articles should be in English language.
- Articles present any material on implementation of agile development practices
- Articles present any challenges in implementation of agile practices especially in large-scale organizations.
- Articles that highlight any visualization technique in agile software development like in project tracking/work status, business and data modeling, requirements tractability etc.
- Traditional industry practices challenges compared with agile practices
- Agile practices challenges along with some solutions are highlighted.
- Articles that are peer reviewed.

2.1.4.2 Exclusion Criteria

- Excluded editorial notes, books and comments.
- Studies didn't provide any qualitative or quantitative data evidence.
- Study that only present author own opinion/experience or lesson learned without any research question and design with concrete evidence.

2.1.5 Study Selection Procedure

For the selection of primary study research literature, the following steps were followed:

- Searched the databases (IEEE, ACM, ISI, Science Direct, Inspec Eng. Village and Scopus)
- Applied basic search filters (Year, Publication, Language, Community, Full Text)
- Identified and removed the duplicates.
- Repetition of studies removed based on Title among the studies
- Relevant studies selected based on Abstract and conclusion of the studies
- Final selection based on the full text review.

2.1.6 Quality Assessment Criteria

By applying quality assessment criteria, the papers for primary studies were selected; those papers had convincing evidence included in primary studies related to the research questions. As suggested by [52], we formed a checklist to figure out the quality of the selected primary study. The evaluation process is presented here;

- Is aim/purpose of the study clearly explained?
- Is the addressed research methodology or approach well defined?

2.1.6.1 Quality Assessment Criteria for RQ1.1

- Is study highlighted any challenges for agile development in large-scale environment
- Are the empirical evidences provided in selected study or not?

2.1.6.2 Quality Assessment Criteria for RQ2.1

- Study addressed different visualization techniques in agile development or not?
- Is there any framework or pattern discussed to solve any challenges in agile development, is it validated in industry or academia?

Along with, the inclusion and exclusion criteria also worked as the quality assessment criteria. Furthermore, for the selection of the primary study, we have given high priority to the papers that have empirical evidence available either in descriptive, qualitative or quantitative form.

2.1.7 Data Extraction Strategy

MS Excel sheets were used to extract data from primary studies. Here is the particular information extracted from primary studies.

General Information

The general information that was extracted from primary studies is given below. These fields are used for determining the year wise publications and the source of the primary studies.

- Title
- Author(s)
- Publication Year
- Source

Specific Information extracted from primary studies for RQ 1.1

- Challenge Reported
- Model of Agile
  - Agile model (Scrum, XP, or general agile practices)
- Type of study
  - Experiment, Survey, case study or interviews
- Type of Challenge
  - General or GSD related
- Solution Proposed
  - Yes or No

Specific Information extracted from primary studies for RQ 2.1

- Visualization Technique reported
- Type of study
  - Experiment, Survey, case study or interviews
- Related to Agile
  - Yes or No
- Related to RE
  - Yes or No
- Area of RE
  - Specification, Elicitation, Prioritization, Management, Analysis, Patterns, Validation, Modeling, Conflicts.

Our motivation for extracting these particular fields from primary studies is to determine which challenges are reported in the literature and these challenges are related to which of the agile practice. We also extracted the type of study that either based on experiment or survey.

2.1.8 Review Protocol Validation

To define review protocol is a critical phase in Systematic Literature Review and even more to evaluate define protocol to fully address research questions [45]. We first started our SLR to define search string to answer RQ1.1 and RQ2.1. For this, we followed guidelines discussed in [45] and following steps to create and later refine our search string;

- Identified search terms in our research questions (RQ1.1 and RQ2.1)
- Get synonyms and alternative words used in RQ1.1 and RQ2.1. We also explore different same domain articles to find controlled terms to minimize any difference in terminology.
- Boolean operators are combined to create a search string.

- Boolean-OR term is used to include synonyms and alternatives terms in our search string.
- Boolean-AND term is use to join key terms that must be present in end.
- Periodic advices from the library personnel for search string refinement and in selection of Search Communities, which is later excluded from our final search string.

For external validation, we had arranged periodic long sessions with library personnel to get continuous feedback and refinement in the search strings. To ensure the internal validity, we presented our search results to our thesis supervisor to get useful feedback and help us to align with our research area. To further create our research in more repeatable form, we also compared our search results with available systematic reviews. Furthermore, to reduce the data extraction threat, we created a spread sheet to maintain complete history/log for our SLR to keep track of search results. This sheet has number of articles found in different search databases with their search results along with proposed solutions.

2.2 Industrial Survey

Survey is one type of empirical study that primarily provides numeric or quantitative description on some sample data collection by asking questions to related peoples[1] or data collection from population of interest [53]. Survey usually structured in the form of standardized questionnaire to grab the required information. This empirical practice facilitate practitioners to gain sufficient knowledge from different participants and in result weight large number of variables [54]. Survey can be used to measure people's perceptions which effectively represent population of interest, knowledge, opinions, and behavioral intentions [55].

Motivation: We carried out survey by creating questionnaire that further strengthens our findings in the literature. Moreover, the need of survey is to get validate literature findings in large scale industrial settings. Our main motivation for going with industrial survey is that it will give us the opportunity to target large number of experts, generalize our findings and increase validations. Survey provides flexibility for participants to provide their responses in terms of project estimated duration, product development approach, development location (co-located or distributed) and success rate of development effort.

Alternatives: Case study or interviews are two alternative research methods of survey. Each of the research method is used differently depending upon the researchers needs. The difference between case study and survey is a matter of degree, and usually involves a balance between the likely generalize information achieved on the one hand, and the details and the likely accuracy of information on specific cases on the other hand [56]. It is not always recognized as we noted earlier is often the survey and case studies presumed distinct types of research, it is sometimes denied that the case studies are intended to be representative or typical in the sense of the survey results. For example, [56] differentiate between the logic "analytical" and "statistical" generalization and argues that only the latter is related to the case studies. Similarly [57], argue as a case study research is "logical" but not "statistically significant" conclusion. Both of these authors characterize generalization from sample to population as "statistical" and unrelated to the case study research.

Case studies are often used to make a thorough study of the phenomenon concentrate on a specific area, often in an industrial environment. Case studies results are difficult to generalize, often partially dependent and can also be used for comparisons.

Another commonly technique used to collect qualitative data is to do personal interviews. Interviews are the best source to collect historical data and collect opinions from the interviewee about particular area to identify potential problems. The aim of

[1] Peoples that associated or involved in agile development cycle

conducting interviews can be to identify the terminology exercised in a particular setting and help to gain in-depth knowledge combination with observation which is usually not possible in survey [58].

Dismissal of Alternatives: About case studies, the main benefit of case studies is that they provide in depth and rich information of a particular area but difficult to generalize the results in other settings and situations. On the other hand, a large number of people from different real situations usually participate in surveys. Therefore, Surveys significantly produced better results in generalization of the gather data with low internal validity as compare to the case study. Online survey questionnaire said to be most efficient way to address large poll of data set in short time period, enable to distribute on multiple channels like research communicates, social forums etc and help to grab more potential respondents [59] [58].

2.2.1 Survey Design

Survey design can be conducted by taking personnel interviews or in the shape of questionnaire. The aim of this survey is to find out different challenges when people practicing agile development in their development effort. Furthermore, if any challenge occurred, how industrial personnel effectively deal with these challenges. More interesting question could be in which way the problem/challenge affecting the development effort and at the end how much efforts needed to mitigate them. In the second part, we interested to see different information visualization techniques in agile development and how these techniques help industry to overcome or reduce the agile challenges.

In survey design, we formed our survey questions in the light of selected primary study and used combination of closed ended and open ended questions to get possible representation of population of interest. In addition, we started with open-ended questions to know participant's area of interest, involvement in agile software development, team structure, project size, development's nature (distributed or co-located), development practice and background information that necessary to generalize the results and more importantly enable us to see agile challenges and visualization techniques in large scale environment. Our survey design consists of two parts; Questionnaire design and samplings. Sampling process is related to the selection of industrial personnel to fill the questionnaire [60]. The design of questionnaire has mixture of open end and close ended question to grab maximum information from the participants and facilitate to find out missing information in the literature.

2.2.1.1 Survey Platform

We selected **Surveygizmo**[2] platform, a customizable online application interface that facilitates us to build web- based questionnaires. It can be used to capture vast range of respondents regard less their presence in collocated or offshore locations. Further motivation behind web based survey is that it only requires a web link to complete the process and therefore speed up the data collection procedure and cost effective technique.

2.2.2 Sampling

Sampling is the procedure by selecting number of respondents in the form of subset from the selected study population. In general, sampling can be performed in two different ways, probability and non-probability sampling. In Probability sampling, systematic approached is used in the selection of subset from total population and samples are produced from the population in shape of statistical diagrams [60][61]. Moreover, in systematic sampling, every participant in the population has equal chance of being selected in the process. In non-probability, this method is applied where individuals of the entire company are not known and then research selects the sample

[2] http://www.surveygizmo.com/

from the population [62]. We selected non-probability approach to carry on our survey. The motivation behind is that, non-probability sampling is used on small scale and difficult to adopt systematic sampling to get large population available in which every person can see statistically equal selection chance [60].

Furthermore, there are two types to perform online survey, Personalized Survey which is our selected type and Self-recruited Survey. In personalized survey, each sample member is known and personally requested in survey participation [63][64]. This type of online survey can be conducted in both probability and non-probability sampling. In self-recruited survey, respondents already know something (over newsgroup, discussion forums) related to survey and eager to actively participate in the survey. We initially personally started with 20 participants that have well experienced in agile development especially on large scale environment and also used any requirement visualization technique or progress tracking technique for their project.

2.2.3 Questionnaires Design

The questionnaires for the survey are designed on the basis of results and findings of systematic literature review. The main aim of the survey is to explore the state of practice and find out requirements visualizations techniques for agile development in large-scale organizations. Therefore in this regards, conducted industrial experts survey to determine whether the challenges of agile development and requirements visualization techniques, reported in the literature can be confirmed by industry experts.

The whole survey is divided into three sections:

- Personal and Organizational Information
- Challenges of Agile Development
- Information Visualization Techniques

It is emphasize on closed ended questions that are used for retrieving information from industrial experts, just because these are easy to answer and provide quantitative information. Also asked the experts to provide extra information on a particular question by designing closed ended questions with “If other, please specify:”

2.2.4 Survey Piloting

The survey design is completed in three steps:

2.2.4.1 Step1

In very first step the results of systematic literature review of both RQ1.1 and RQ2.1 were reviewed. On the basis of results and analysis of both systematic literature reviews the initial questions were designed. These questionnaires were designed by emphasizing on close-ended question with option for experts to enter any other if suitable that particular case.

2.2.4.2 Step2

After designing the initial questionnaires of the survey, it was sent to fellow students of software engineering, who have experience in agile software development. On the basis of their feedback the questionnaires were improved. Those questions were modified or removed that were creating confusions or ambiguities. Also the suggestions and ideas from students were entertained and placed in the survey.

2.2.4.3 Step3

At the end after applying all modifications suggested by fellow students, the updated survey questionnaires sent to closely attached industrial experts for verifications. Implemented changes and modified questionnaires on the basis of feedback from closely attached industrial experts. Once all necessary changes were implemented then final questionnaires sent to industrial experts.

2.3 Grounded Theory

Grounded Theory (GT) defined by Glaser and Strauss [65] [66], was used to analyze systematic literature review and survey results. It is qualitative systematic research approach, initially designed for social sciences. In GT, a theory is generated on the basis of raw data by using bottom up approach. Initially raw data is coded into different categories (open, axial or selective coding) and analyzed to generate a theory. Through GT researcher can constantly make comparison between collected data. In GT, analysis can be done by following predefined procedure of GT by applying open, axial, and selective coding techniques on collected raw data [67].

Motivation: Our motivation for selecting Grounded Theory for analyzing results of this study is because it is a suitable technique to build understanding of raw data collected from literature review or through industrial survey. Through GT we can analyze raw data rather than relying on existing concept or "off the shelf" theories [68]. Second advantage of using GT is by using we can introduce new concepts during analysis phase. In addition to this, GT does not restrict that all data points are to be collected before starting the analysis process. It allows us to add not only the data points but also add concepts at any stage of the analysis process [69].

Alternatives: For alternatives of grounded theory, we can use interpretive techniques like observer impression or recursive abstraction. In observer impression, an expert or bystander observer examines the research data, interpret it in the form of an impression and format the impression in a structure form. While in recursive abstraction datasets are summarized; these summarized datasets are then further summarized and so on. At the end result is generated in more compact summary that would be used to generate results from datasets [68].

Dismissal of Alternatives: For analysis of results found through systematic literature review and reported by industrial experts, we have to use early data analysis method. Grounded theory, one of early data analysis method, provides the facility to build understanding of raw data collected through both processes. While the limitation of alternatives are; for using observer impression, there would be an expert required who can interpret the datasets, while for recursive abstraction there are highly chances that final conclusion misses several important dataset. By using grounded theory we can add more datasets and can introduce new codes at any stage of analysis [69].

2.3.1 Application of Grounded Theory

Grounded Theory was used for analyzing qualitative data collected through system literature review and industrial survey. By using GT, we applied codes and categories to collected data and analyzed it for generalizing our results. This approach facilitates us to categorize the collected data according to their specific area and helps to analyzed it with better understanding. Following steps were performed for applying grounded theory on collected data.

2.3.1.1 Step 1: Open Coding

For application of Grounded Theory the first was performed to apply code on every point of collected data. The codes were assigned as much closed meaning to raw data collected through systematic literature review and industrial survey [68]. These codes were assigned so that we can analyze the collected data.

2.3.1.2 Step 2: Axial Coding

The next step in applying grounded theory on collected qualitative data is applying axial coding on open codes of first step. Axial coding is applied on the basis of interrelated and interlinked open codes with each other. Axial coding helps to understand the collected data more precisely [70].

2.3.1.3 Step 3: Selective Coding

In selective coding step of applying grounded theory, axial codes are analyzed and categorized into more general categories to understand the actual data [70]. Selective coding helped us to categorized challenges into generalized form so that we find solution of these challenges by applying visualization techniques.

2.3.2 Use of Grounded Theory in Analyzing Results

For this particular research study, we used grounded theory not only for analyzing industrial survey data, but also on analysis of SLR data. Grounded theory provides the systematic steps to analyze the qualitative data, that's why we were motivated for the application of grounded theory to analyze data. Data extracted from literature review and reported in industrial experts was assign codes (open coding). These codes were then grouped into related concepts (axial coding). At the end related concepts are finally grouped into categories (selective coding). A typical example of application of grounded theory is given below.

Initial Extractions (Open Coding):	**Concepts (Axial Coding)**	**Categories (Selective Coding)**
In highly hierarchical companies, the figure of product owner is assigned normally a person with managerial responsibilities which lacks in time to actually comply with the product owner's tasks. This forces the team to adapt the methodology to the conditions, but the requirements handling (definition and prioritization mostly) becomes severely affected. It is easy to realize this problem when the team uses the sprint planning meeting to define requirements and also estimate the times.	Requirements handling (Definition and Prioritization)	Requirements engineering
Design knowledge remains with the individuals. When implementing a feature, nobody but one person seemed to know the difference between two different modes of performing a task. The business need for both could not be found and the only resource with certain knowledge was the application architect Increased maintenance effort with increase of the number of releases. The addition of new features makes the maintenance process more complex.	Design knowledge remains with the individuals	Communication, Coordination and Control

Table 4: Example of grounded theory application

3 RESULTS AND ANALYSIS OF SYSTEMATIC REVIEW

Two different SLRs were performed in order to find out the challenges of agile in large-scale organizations and Information visualization techniques in current state of art. For presentation of the results, tabular and graphical charts were used.

3.1 RESULTS AND ANALYSIS OF SYSTEMATIC REVIEW for RQ 1.1

3.1.1 Selection of Primary Study

For the selection of primary study for RQ1.1, the procedure mentioned in section 2.1.6 was followed.

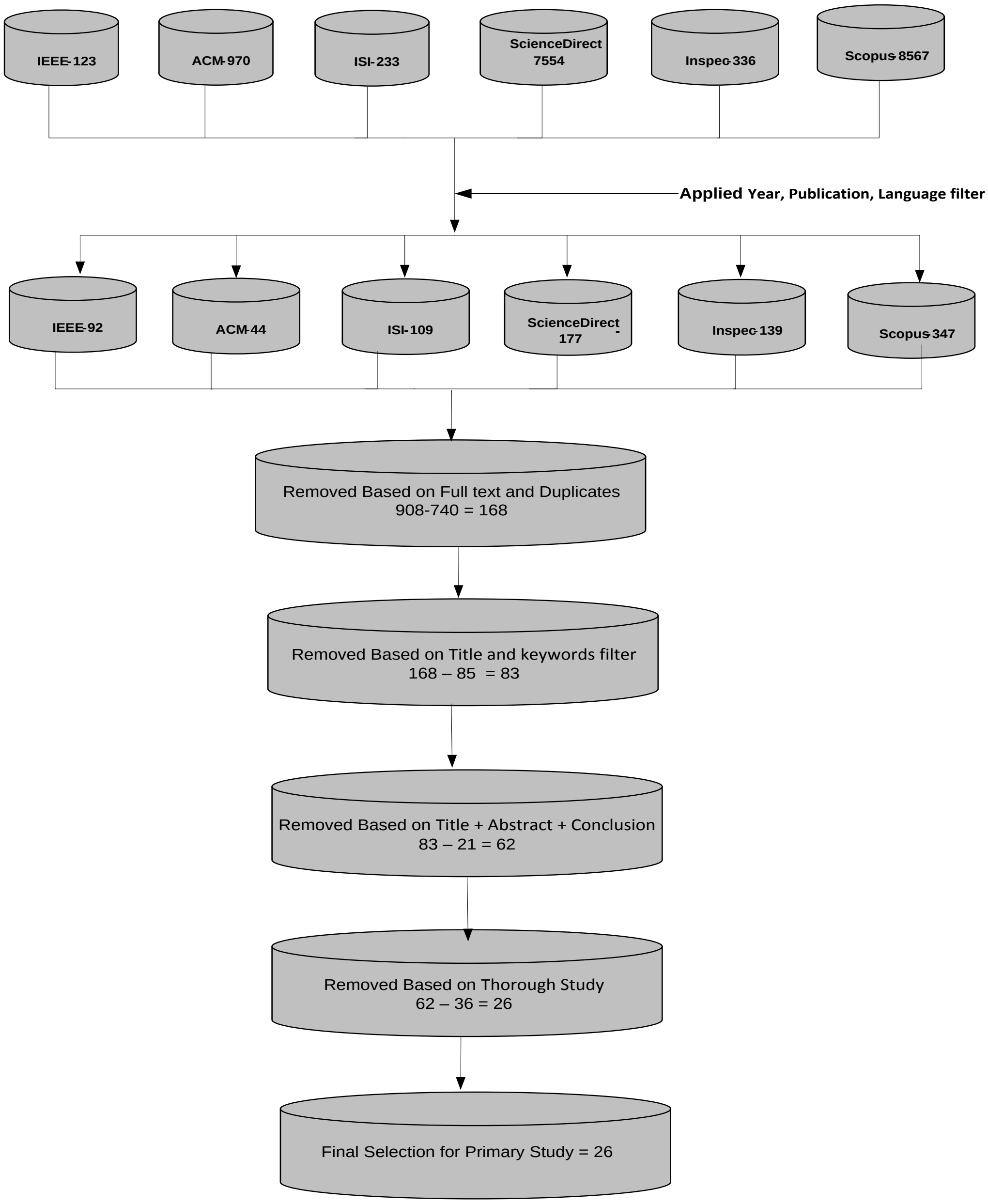


Figure 2: Selection Procedure of Primary Study for RQ 1.1

3.1.2 Classification of Included Studies

For the primary study of RQ1.1, there were 26 research articles were selected. Although for the selection of primary study, time period was between 2000 and 2011, but after applying inclusion/ exclusion criteria the articles were between 2005 and 2011.

Year	Number of Articles
2005	2
2006	4
2007	2
2008	4
2009	8
2010	5
2011	1

Table 5: Selected Study by Years for RQ 1.1

Among 26 selected research articles, 14 related to general agile practices while 9 were from Scrum and 3 from Extreme Programming (XP).

Study Area	Number of Articles
Scrum	9
XP	3
General	14

Table 6: Selected Study Areas for RQ 1.1

The majority of selected primary study consists of Case Studies with count 12 from 26. There were 4 articles based on literature review, while 4 based on Survey and 6 based on Interviews.

Study Area	Number of Articles
Interviews	6
Surveys	4
Case Studies	12
Literature Reviews	4

Table 7: Selected Study Types for RQ 1.1

3.1.3 Summary of Challenges

Here is the summary of challenges for agile development found in primary study.

Challenge	Description	Agile Practices	Reported
Communication, Coordination and Control	• Communication and cultural challenges • Communication among developers and customer • Customer access and participation • Design knowledge remains with the individuals • Collaboration between Cross-team and within team members • Slow Decision-making (many layers of customer and stakeholders)	• Scrum • XP • General Practices	[26], [48], [71], [72], [73], [74], [75],[76], [77], [78], [79], [80], [50]

Requirements Engineering Challenges	• Requirements engineering problems • Neglect of non-functional requirements • Prioritization on a single dimension • Inadequate requirements verification • Misunderstanding requirements. • No focus on quality attributes • Long RE duration due to complex decision processes	• Scrum • General Practices	[48] , [81], [72], [71], [75]
Process Conflicts	• Peoples Conflicts (related to team and upper management resistance and transition issues) • Development process conflicts (Lifecycle, development processes, Functional n non-functional requirements) • Business process conflicts (organization standards, conformance of traditional process standard)	General Practices	[72]
Adoption of Agile Development	• Integration of agile projects with existing processes • Automation is critical with existing processes • Lack of Tool Support • Large Number of Project Personnel • Lack of Collaborative Office Environment • Area product owners • Common sprint planning • Scrum-of-Scrums • Common sprint demo • Common retrospective • Corporate Governance (scaling Scrum to entire company)	• Scrum • XP • General Practices	[82], [83], [76], [84], [79], [85], [86]
Architectural Issues	• Inadequate or inappropriate architecture • Architecturally very risky for new projects • Lack of focus on architectural design	• XP • General Practices	[48], [71], [72], [78]

Configuration Management	• Configuration management and version management • Minimal documentation • Increased maintenance effort with increase of the number of releases • Increase in product packaging effort	• Scrum • General Practices	[48], [73] [76] [87]
Software Quality	• Continuous refactoring clashes existing quality control systems • Hard to produce quality software • Realize continuous testing • Reduction of test coverage due to shortage of projects timeline • Lack of independent testing	• Scrum • XP • General Practices	[48], [74], [83], [88]
Management Issues	• Creating and managing the requirements analysis teams • No time for careful design during iterations • Flow Leveling for Limited Resources • Team management • Management overhead	• Scrum • General Practices	[48], [81], [72], [84], [86], [50]
Lack of Measurement Metrics	• Problems with cost and schedule estimation • Net Present Value Estimation • Measuring Improvement (e.g. productivity) • Measure Quality factors • Difficult to judge and Map productivity on some metrics	• Scrum • General Practices	[71] , [76], [80] , [86]

Table 8: Summary of Challenges found in literature

3.1.4 Challenges of Agile in Large-Scale Organizations

There are many challenges reported in the current literature in shape of communication, coordination, collaboration, architectural issues, process conflicts when development teams working in co-located or in distributed/offshore settings simultaneously. For this particular study, we selected all sort of challenges that are related to agile development in large-scale organizations. These challenges may vary their impact on the project or organization. The main motivation for finding all sort of challenges is to get aware that how different challenges are reported in literature. The other motivation was building a baseline for survey questions so that we can familiar with all sorts of challenges. These challenges are elaborated below.

3.1.4.1 Communication, Coordination and Control (3Cs)

While studying the primary literature, it is identified that the communication, coordination and control (3Cs) related challenges are reported as main challenges [48], [26], [71], [72], [73], [74], [75],[76], [77], [78], [79], [80].

3.1.4.1.1 Communication and Cultural Challenges

Most of the time researchers identified that when there are large teams working on distributed environment then they faced communication is their major challenge. In paper [73], mentioned the challenges related to communication and cultural differences that caused major issues for scaling agile in distributed software development environment. Communication gap between the development teams and customers also cause hindrance in the success of software projects [26]. Papers [75], [78], [79] also identified the challenges related to communication richness and lack of coordination among distributed development teams.

3.1.4.1.2 Knowledge Sharing and Collaborations

Lack of group awareness, cross-team collaborations and collaboration within team members are the challenges highlighted by [74], [72], [76], [77] and [79].

3.1.4.1.3 Stakeholder's Participation

When teams are working in distributed environment, it's hard to involve all stakeholders in every phase of the software development. This issue causes the delays and creates overhead [48], [71] and [80]. Time increases between different phases of development processes like design phase wait for requirements phase to finish.

3.1.4.2 Requirements Engineering Challenges

Requirements engineering is one of the key process area of software development. Papers [48], [81], [71], [72] and [75] have highlighted the challenges related to RE practices.

3.1.4.2.1 Elicitation, Specification and Prioritization

Requirements elicitation process may take long requirements engineering duration due to complex decision processes [48] and often misinterpreted requirements can lead to failure of software project [75]. Requirements priority lists are hard to create and maintain because prioritization is done on single dimension [48], [81] and [71].

3.1.4.2.2 Validation and Quality Attributes

Software quality attributes (non-functional requirements) often neglected in large-scale requirements engineering process [71], [72]. Requirements validation is hard to achieve with absence of non-functional requirements.

3.1.4.3 Process Conflicts

When large teams working in distributed environment may face challenges related to their process conflicts [89]. These conflicts can be related to development, business or people associated with the processes.

- ***Development process conflicts***: Legacy systems, Different life cycle, Requirements etc.
- ***Business process conflicts:*** Progress management, human resource, Process standard rating.
- ***People conflicts:*** Management issues, logistical issues, change management, handling successful pilots.

3.1.4.4 Adoption of Agile Development

In large-scale distributed environment, it's a challenge to adopt agile development because there is lack of tool supported, large number of project personnel and increase number of sites, and lack of collaborative office environment [82], [83], [76], [84], [79], [85], [86]. Automation is critical in all management phases of software development like configuration management, build management, quality management, change management/defect tracking, planning and scheduling [84]. If we talk specifically about Scrum, the challenges reported are related to non-availability of area product owner in distributed environment, common sprint planning, Scrum-of-Scrums and common sprint demo [76], [85].

3.1.4.5 Architectural Issues

In large-scale software project environment, number of challenges reported related to inadequate or inappropriate architecture of software [48], [71], [72] and [78]. These challenges are arises when possible solutions are not very well understood or clearly stated in the start of the projects and in result create poor architecture design [72].

For handling architectural related issues, [71] proposed a framework to overcome architectural issues of software in large-scale environment illustrated in Figure 3.

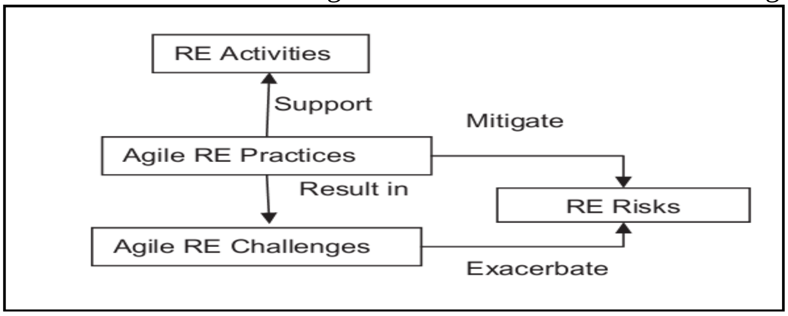


Figure 3: Agile RE framework [71]

3.1.4.6 Configuration Management

Papers [48], [73], [76] and [87] reported the challenges related to configuration management and release planning, when number of releases increases then it also increases the maintenance efforts as well as flow leveling for limited resources. The typical challenges are;

- Level the flow
- Backlogs priority
- Release planning
- Time to market

3.1.4.7 Software Quality

In large-scale software development environment, pair programming is difficult to do formal reviews. Papers [48], [74], [83] and [88] reported the challenges associated with continuous refactoring clashes to the existing quality control systems, lack of independent testing also creates problems and in result heavily depend upon continuous testing in large-scale distributed environment.

3.1.4.8 Management Issues

In large-scale software development environment, management issues have always a large influence on the performance of teams [48], [81], [72], [84] and [86]. Paper [48], reported that continues coordination between teams is required to overcome challenges related to communication and coordination. However, this continues coordination is itself an overhead to the management of the organization. Creating and managing the requirements analysis teams is also an issue related to the management of agile teams [84]. Managing teams in large-scale organizations can create governance problems and difficult to have flow leveling of limited resources [81] and [86]. Time management has also play a vital role in large-scale agile software development environment where there is no time for careful design during iterations [72].

3.1.4.9 Lack of Measurement Matrix

Measurement of team productivity, cost, and time or software quality factors, in large-scale organizations is a challenging task [71], [76], [80] and [86]. It is difficult to judge and map productivity on some metrics and measure software quality factors in large-scale traditional organizations [80]. There is a need to improve measurement matrixes to measure different factors like productivity, cost and time [76], [86].

3.1.5 Analysis of Results for RQ1.1

The systematic literature review for RQ1.1 was performed to find out the challenges of agile development in large-scale organizations available in current literature. Challenges of agile are highlighted in two main areas of software development in selected research articles. These areas are agile in large-scale organizations for distributed and co-located environment.

The challenges reported in selected studies are related to communication, coordination and control, requirements engineering related challenges, scaling of agile in large-scale, management of resources, lack of measurement metrics, configuration management and architectural issues. The selected studies were covered mainly general agile practices while 9 and 3 studied articles were related to Scrum and XP practices of agile. One of the main emphasize during selection of primary study articles was to get those articles that should have some empirical evidence or review of literature. Through reviewing current literature, the most cited challenge current industry facing is to create and maintain continuous communication and coordination bandwidth in between development teams, and to get strong control over geographical dispersed teams and co-located teams in large scale development. Another strong challenge is when scaling agile development over large organization environment: people who are already familiar with the current system usually resist to any change in the current system and these behaviors create more problems in the process of scaling agile. Secondly, scaling agile is complex process especially on large scale environment and need special attention to understand what is going on in the current system and how to deal with existing development processes. On the other hand, small projects have much better control and in result can provide better traceability in their development processes. In addition, given table 9 gives a bird eye view of the challenges found in the literature with respect to their impact.

Challenge	Found
Communication, Coordination and Control	13
Requirements Engineering Challenges	5
Process Conflicts	2
Adoption of Agile Development	8
Architectural Issues	4
Configuration Management	4
Software Quality	4
Management Issues	6
Lack of Measurement Matrix	4

Table 9: Challenges found in SLR

3.1.6 Comparison with Previous SLRs

The systematic literature review by Wohlin et al, [48], conducted a case study in large scale development environment to highlight issues and challenges, and compare them with the previous empirical studies. The main objectives of this study were to exemplify the way to implement agile and incremental practices in large scale organization environment; provide in-depth knowledge/understanding of issues and advantages related to agile development and at the end generalizing own findings and compare them with state of the art literature. In the systematic review conducted by Dybå et al [49], no selected study had clear direction or aim to identify issues and challenges using agile development. Although, study identified number of limitations and benefits of using agile development but evidence strength is quite low that makes it more challenging task to propose any solution to the agile industry.

In context to our research, we are more interested to highlight different challenges faced by large industry using agile practices in both co-located and distributed development settings. Moreover, the selected previous systematic reviews cover the study period till 2009. Our study highlighted some new challenges in shape of continuous refactoring clashes existing quality control systems, reduction of test coverage, flow leveling of limited resources and some more highlighted in Table 10, that are missing in the previous systematic literature reviews (SLR). However, most of the challenges already well addressed in [48] review that gives strength to our study and provide validity to our findings in current literature.

	Previous SLRs	Our SLRs
Area of Agile	Mostly on Scrum and XP	General
Development Environment	Co-located	Both Co-located and Distributed
Years Covered	Till 2008-09	Till 2012
Team Size	20-30	More than 50

Table 10: Comparison with Previous SLRs

3.2 RESULTS AND ANALYSIS OF SYSTEMATIC REVIEW for RQ 2.1

3.2.1 Selection of Primary Study

For the selection of primary study, the procedure mentioned in section 2.1.6 was followed.

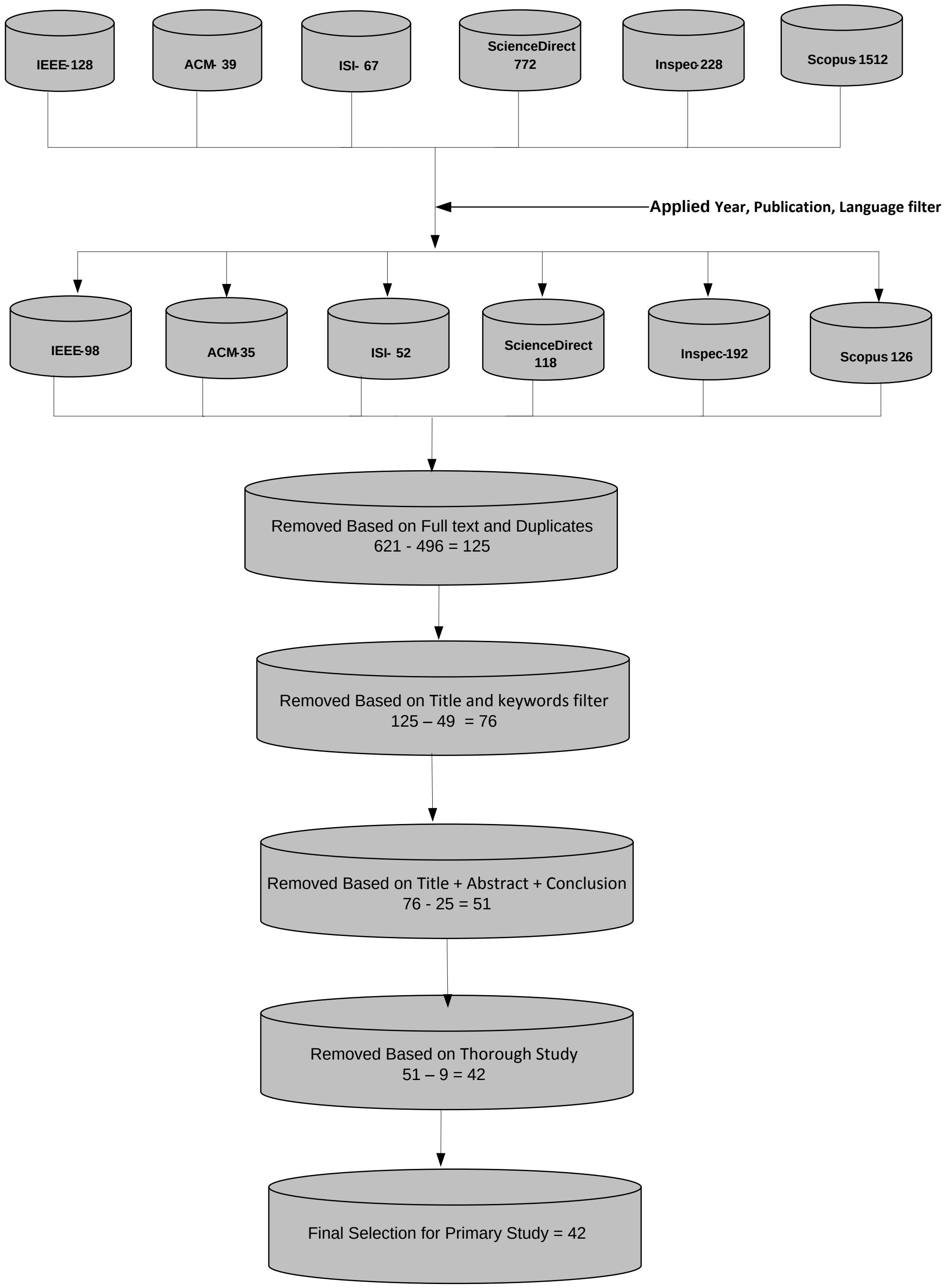


Figure 4: Selection of Primary Study for RQ 2.1

Total number of identified primary studies for RQ2.1, are 42 research articles that fulfill the inclusion and exclusion criteria.

3.2.2 Classification of Included Studies

For primary studies of RQ2.1, 42 research articles were selected. Articles were selected between year 2000 and 2011. The most number of articles found with 9 in year 2006 and found 7 articles in 2009 while 6 in 2008.

Year	Number of Articles
2000	2
2001	2
2002	3
2003	0
2004	1
2005	3
2006	9
2007	3
2008	6
2009	7
2010	4
2011	2

Table 11: Selected Study by Year for RQ 2.1

The majority of selected primary study based on Case Studies with count 20 from 42. There were 8 articles based on experiment, 3 based on Framework proposed and 5 based on Tool designed and on literature review with 1 based on interview.

Study Area	Number of Articles
Case Study	20
Experiment	8
Framework Proposed	3
Interviews	1
Literature Review	5
Tool Designed	5

Table 12: Selected Study Types for RQ 2.1

3.2.3 Summary of Visualization Techniques

Category	Visualization Techniques	Description
Project tracking and work status	Scrum Task Board	Team members meet in front of a task board and update their status of work. The task board captures what is called the sprint backlog.
	Cumulative Flow Diagrams (CFD)	Cumulative Flow Diagrams are another way for tracking agile project progress that building upon the basic Burn-Up charts.

	Parking lot Charts	Summarize the top-level project status. It is first described in Feature Driven Development (FDD), sometimes also called a “Project Dashboard".
	Burn down Charts	Count the number of Kanbans (backlog tasks) and track it in a time box to show the trend of work accomplished.
	Scheduling With A Vision (SWAV)	SWAV is designed to help project managers to identify conflicts during the entire scheduling period. It should also provide intermediate levels of understanding about the resource availability/shortfall situations.
	Team Calendar	The team calendar is also a good place to write down any team events, indicate start and end of sprints etc.
	Strawman	Technique provide way to associate program RE to project RE means map program phases to project phases in different time frames.
Business Modeling	Data Flow Diagrams (DFD)	Show the flow of data from external entities into the system, show how the data moved from one process to another, as well as its logical storage.
	Flow Charts	Flow charts are typically used to describe the detailed logic of a business process or business rule.
	Unified Modeling Language (UML)	SysML, Unified Model of Dependability (UMD), URML, StarUML, REVU, UML Case.
	State Charts/State Machines	Similar to UML.
	Use case diagrams	A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse.
Data modeling	2D/3D Visualization	These visualization techniques help validation of requirements reducing the communication gap between the customers and the developers. The aim is to improve the efficiency of the requirements validation process (Requirements are elicited and then

		documented).
	Mondrian	The primary focus of this visualization approach is to offer the programmer the possibility of visualizing his data model while using his preferred environment and tools.
	TreeMap	During requirements gathering, visualization is used to help scrutinize the status (completeness, extent) of the information.
Requirements analysis and traceability	Graphics Visualization	Presented graph based visualization to show requirement analysis and management. In addition, explore relationship between requirements to give better understanding of requirements dependency or traceability.
	Sunburst, Netmap	Sunburst and Netmap, visualization techniques are adopted for representing traceability links between requirements knowledge.
Software Configuration management	PlasticSCM, Revision tree	An interactive 2D visualization, named Revision Tree which allows visualizing the contributions of the team members, through several revisions, baselines and long periods of time, on the same item or document within the software project.
Coordination among development teams and project stakeholders	Scrum storyboard	In Scrum, the “Scrum of Scrums” is a way to ensure alignment and coordination across different teams, or among different sub-teams of a large Team (colored Teams).
	Road Maps	The Roadmap another information visualization technique consists of a series of planned activities and their dates, each of which has a theme and a prioritized feature set.
	User Story Mapping	A user story map arranges different user stories into useful/appropriate model that help to understand the system functionality, identify omissions and holes in project backlog and efficiently plan holistic releases that gives value to business and user with each release.

Table 13: Summary of Visualization Techniques

3.2.4 Information Visualization Techniques

After in depth studying primary study, here is information visualization techniques found those can be applied in agile development.

3.2.4.1 Business Modeling

Business modeling is the one area in which lots of different techniques and tools are available for representation of business models. Data Flow Diagrams (DFD), Flow Charts, Unified Modeling Language (UML), State Charts/State Machines, Use case diagrams and Feature Transition Charts (FTC) are the main example of these techniques and tools [90], [91], [92], [93], [94] [95], [96], [97], [98], [99], [39] and [38].

3.2.4.1.1 Data Flow Diagrams (DFD)

Data flow diagrams can be used for representation of requirements elicitation and validation. The paper [92] emphasized on security requirements and how security requirements can be traced out by using data flow diagrams.

3.2.4.1.2 UML

Unified Modeling Language is one of the main sources of visualizing software business models; most of the researchers either used or recommend this language to represent the requirements [91], [92], [95], [96], [97], [98], [99], [39] and [38]. It can be used for requirements elicitations (to find the traceability in requirements), specification and analysis in the form of visualized representation of requirements. Paper [95] mentioned that UML can be used for presenting multiple components (attributes) of requirements.

3.2.4.1.3 State and Feature Transition charts

State charts are similar to UML and can be used for visual representation of requirements specification. The paper [90] describes that how state charts can be used for visual representation of requirements specification in hierarchal decomposition, synchronization and in concurrent form. The paper [93] describes that how Feature Transition Charts provides the scope overview of decisions that involve chains in multiple projects based on previous experience.

3.2.4.2 Project Tracking and Scheduling

There are number of Information visualization tools and techniques are used for project tracking and scheduling in agile development. These tools and techniques are Scrum Task Board, Parking lot Charts, Burn down Charts, Scheduling with A Vision (SWAV), Team Calendar, Strawman, Cumulative Flow Diagrams (CFD) and User Story Mapping [100], [33], [34], [101], [102], [21], [31], [30], [29].

In agile team members meet in front of a task board and update their status of work. The task board captures what is called the sprint backlog. Paper [34] uses Natural Scene Paradigm to support daily Scrum meetings and highlight change structure of tasks in natural language. Parking lot Charts summarize the top-level project status. It is first described in Feature Driven Development (FDD), sometimes also called a “Project Dashboard" [102].

SWAV helps project managers to identify conflicts during the entire scheduling period in agile development. Paper [21] contributed in the domain of SWAV by facilitating project managers to effectively deal with uncertain resource constraint scheduling and management. The team calendar is also a good place to write down any team events, indicate start and end of sprints etc. Paper [101] presents that how team calendar gives values to manage team and resources in a better way by adding team visibility to plan bit in advance.

Burn down Charts counts the number of Kanbans (backlog tasks) and track these Kanbans in time box to show completed work. In the paper [33] uses Burn Down Charts for progress tracking and show the remaining work with respect to time frame.

3.2.4.3 Requirements analysis and traceability

Different visualization tools and techniques are used to represent requirements analysis and traceability in visualize form. Chain Graph, Sunburst and Netmap are the example of tools and techniques used to represent requirements analysis and traceability [97], [103], [104], [105] and [106].

Graphs can be used for tracing and structuring different requirements, Chain Graph presents the relationship of requirements to explore dependability and traceability of requirements [107], [103] and [97]. Sunburst and Netmap are the tools used for representing traceability links between requirements knowledge [104].

3.2.4.4 Data Modeling

2D/3D Visualization (ReqViz3D, linegraph, Scatterplot, Chernoff faces, RadViz, XGobi, PlasticSCM), Mondrian and TreeMap are the tools and techniques that are used in data modeling to visualize requirements. These visualization tools and techniques help requirements validation by reducing the communication gap between the stakeholders [108], [109], [38], [110], and [32]. Paper [32] proposes a framework that focuses visualization approach to provide facility for programmers the possibility of visualizing their data model while using their preferred environment and tools. TreeMap [108] used during requirements gathering process, visualization is used to help scrutinize the status (completeness, extent) of the information.

3.2.4.5 Coordination among Stakeholders

The “Scrum of Scrums” is used in Scrum to ensure team’s work alignment and coordination across different development teams. It can also be used among different sub-teams of a large Team (colored Teams) [111]. StoryBoar and Roadmap provide different ways to support a mutual understanding and coordination among internal, external stakeholders and planned or proposed product releases[112], [113] and [114].

3.2.4.6 Miscellaneous

Different other visualization tool and techniques are also reported in literature. These tools and techniques are Distributed Collaboration Priorities Tool, RE Animator, Class Model, Wordle tool, IRIS tool and Metis Tool [91], [115], [116], [117], [118], [119] and [38]. These tools and techniques are used for information visualization in software development.

3.2.5 Analysis of Results for RQ2.1

These days business stakeholders want their applications functioning well and behave according to market trends. They setup their application features according to the system needs and address well to gear up their day to day organizational processes. The systematic literature review for RQ2.1 was performed to find out different information tools and techniques used in general for software development process, product development and tracking activities in agile development. In literature review, we found number of visualization techniques that can be used for business modeling, project tracking and scheduling, requirements analysis and traceability, data modeling, and coordination among project stakeholders.

In business modeling, we experienced that most of the projects are using unified modeling language (UML). It offers standard ways to construct or visualize software architectural elements. It can be used to visualize project activities, business processes, programming models and hence cover all the processes in software development life cycle. We found lot of different modified versions of UML in current literature like SysML, StarUML, REVU, UML Case etc. that can be used for business modeling.

In project tracking and scheduling, number visualization tools and techniques are reported in context with agile development like burn down charts, Scrum task board, team calendar, parking lot charts etc. The Scrum task board is most cited in the current literature, in which team members can organize themselves and meet in front of task board to update their working status. This exercise helps project stakeholders to quickly jump to current project status (in terms of Sprint backlog) to know how much work is left in the project and at the end gives good view of time estimation to complete the project. In data modeling, different visualization tools and techniques reported in the literature like different tools in 2D/3D visualization technique e.g. ReqViz3D, linegraph, scatterplot, Cherno faces, RadViz, XGobi, these visualization tools help in validation of requirements, reducing the communication gap between the customers and the developers. The aim is to improve the efficiency of the requirements validation process.

On the other hand, Scrum Task Board and Burn down Charts used in agile software development where team members meet in front of a task board and update their status of work. The task board captures what is called the sprint backlog. A user story map, another information visualization technique arranges different user stories into useful/appropriate model that help to understand the system functionality, identify omissions and holes in project backlog and efficiently plan holistic releases that gives value to business and user with each release.

4 RESULTS AND ANALYSIS OF INDUSTRIAL SURVEY

4.1 SURVEY RESULTS

Online survey source Surveygizmo, used to collect responses from survey respondent. Appendix – has details of survey questions.

For contacting industrial experts, we had used these main channels;

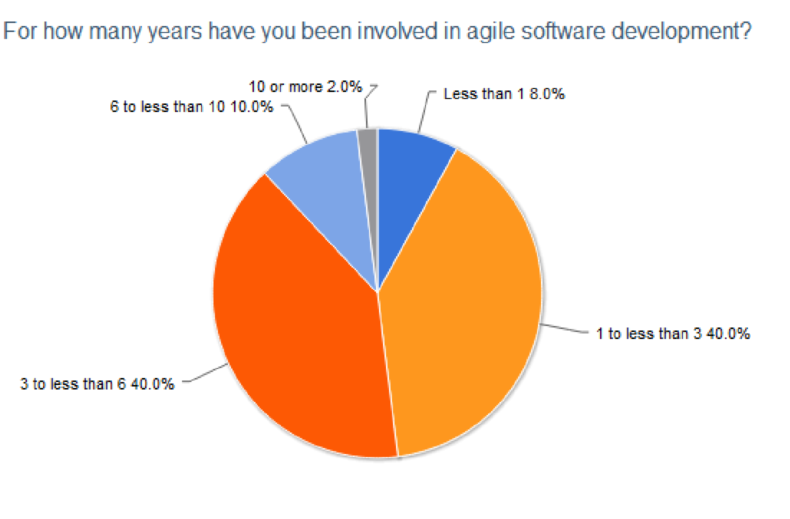
- Emails (Personal and Professional Contacts)
- XP2012 Conference [3]
- Agile Groups and Forums
- Social Networks
- Personal Visits

We had received 125 responses from industrial experts, 49 were completed responses while 74 were somehow partially completed. After analyzing those 74 partially completed results, we included 1 more response for challenges of agile and for information visualization techniques.

Here are few demographic questions and their responses.

4.1.1 Experience

Asking about experience of working in agile environment allowed us to know that how reliable the answers of those respondents are. About 80% of survey respondents have experience of 1 to 6 years working in agile environment. Only 4 respondents have less than 1 year of experience so we can say that most of our survey respondents have well experience in the field of agile software development and they are aware of challenges of agile and familiar with information visualization techniques.



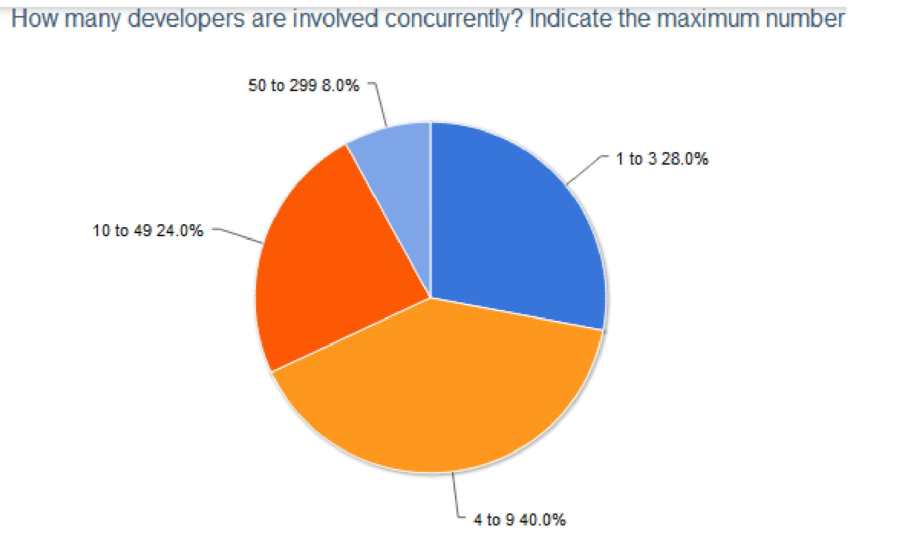
Value	Count	Percent %
Less than 1	4	8%
1 to less than 3	20	40%
3 to less than 6	20	40%
6 to less than 10	5	10%
10 or more	1	2%

Figure 5: Years involved in Agile Development

[3] http://xp2012.org/

4.1.2 Team Size

Asking about team size that how many developers are involved concurrently allowed us to aware about the team size and is there any impact of team size on challenges of agile development. Large number of survey population working in small teams i.e. 1 to 9 people working concurrently to be précised 68% of respondent working in small teams of 1 to 9 developers. While 32% of respondent working in large teams i.e. there are more than 10 people working in a team.

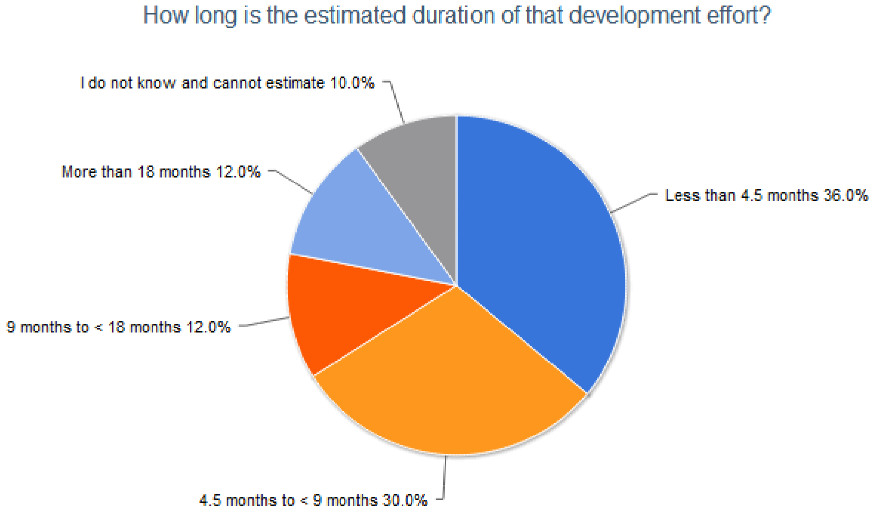


Value	Count	Percent %
1 to 3	14	28%
4 to 9	20	40%
10 to 49	12	24%
50 to 299	4	8%
300 or more	0	0%
I do not know and cannot estimate	0	0%
Cannot estimate	0	0%

Figure 6: Team Size

4.1.3 Duration of Development Effort

Asking about development duration allowed us to know the development duration and compare its impact different challenges of agile development according to duration of development effort. About 66% respondent mentioned that they are working on medium or small projects and their duration is 9 or less than 9 months.

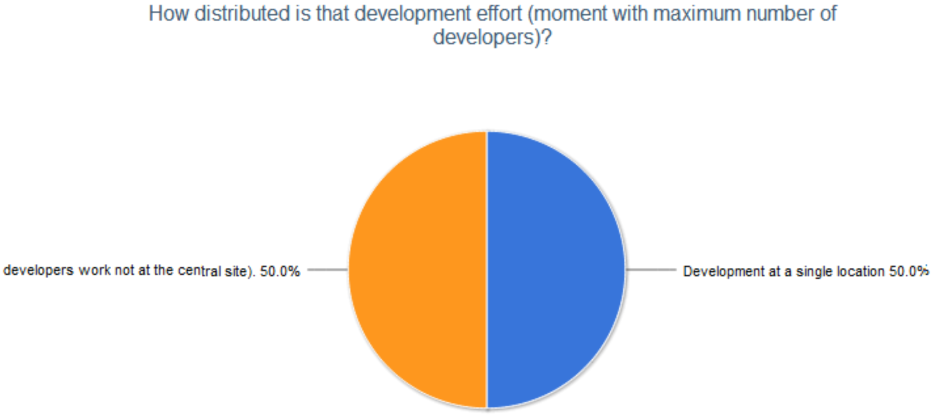


Value	Count	Percent %
Less than 4.5 months	18	36%
4.5 months to < 9 months	15	30%
9 months to < 18 months	6	12%
More than 18 months	6	12%
I do not know and cannot estimate	5	10%

Figure 7: Estimated Duration of Development Projects

4.1.4 Development Environment

Asking about development environment allowed us to aware that either the development environment means how co-located or distributed environment affects the challenges of agile development or not. We had received equal amount of respondents from both environments with 50% at single while same 50% at multiple locations.

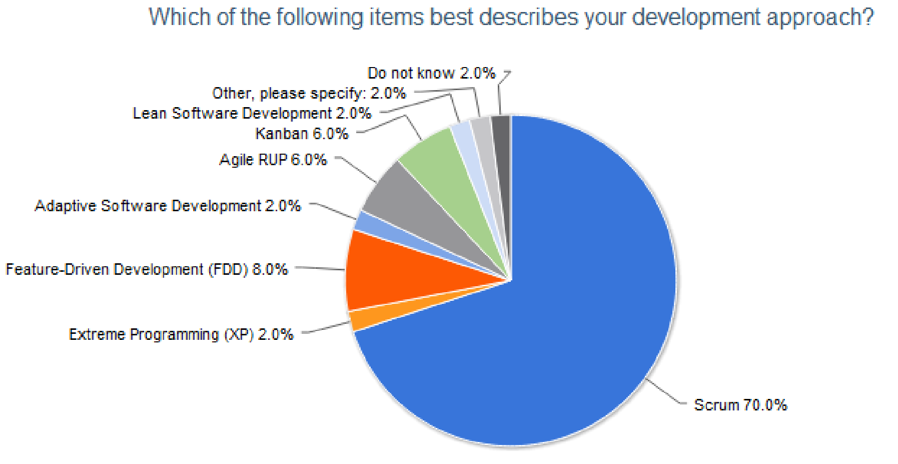


Value	Count	Percent %
Development at a single location	25	50%
Development at multiple locations (at least 10% of the developers work not at the central site).	25	50%

Figure 8: Development Environment

4.1.5 Agile Practice

Asking about agile practice allowed us to aware that how different agile practices affect on challenges of agile development and which information visualization technique is mostly used to avoid or mitigate those challenges. About 70% of our survey respondents were practicing Scrum as agile methodology.



Value	Count	Percent %
Scrum	35	70%
Extreme Programming (XP)	1	2%
Dynamic Systems Development Method (DSDM)	0	0%
Feature-Driven Development (FDD)	4	8%
Adaptive Software Development	1	2%
Open Unified Process (Open UP)	0	0%
Agile RUP	3	6%
Kanban	3	6%
Lean Software Development	1	2%
Crystal Methods	0	0%
Other, please specify:	1	2%
Do not know	1	2%
Behavior Driven Development	0	0%

Figure 9: Agile Practice

4.2 CHALLENGES OF AGILE

4.2.1 Communication, Coordination and Control (3Cs)

Under this concept, challenges related to communication, customer access and participation, knowledge sharing and team collaboration issues involved in large scale agile development.

From the results of our Survey, ten respondents strongly addressed this issue in their development teams especially in remote teams and client meetings. One survey respondent experienced that most of the time team members located at different location suffers from control problems, which is somehow due to unstable coordination. Since their product is domain oriented so most of the time they need to visualize the actual or reflective behavior of product from customer perspective, which is tricky thing to justify the requirements. However there is one more attribute underlying this whole challenge and that is a design artifact or analytic ideas remains to individual or group of team members, so if any fluctuation in coordination occurred that indirectly effects control on development/integration process.

Another challenge in this concept more strongly highlighted by the respondents is the slow decision making process. There were many layers of stakeholders and customers end to make decision on system requirements and design artifacts of the system. This challenge is also reported due to complex hierarchy of the organization. In Scrum development, where sprints size suppose to be 3 to 4 weeks long but due to time taking or slow managerial process, sprints size per long 5 to 6 weeks and in result gives strong effect in development process and product release planning.

Moreover, another challenge reported in the form of design knowledge remains with the individual in agile development. When implementing a feature, nobody but one person seemed to know the difference between two different modes of performing a task. The business need for both could not be found and the only resource with certain knowledge is the application architect. This challenge is in result increased maintenance effort with increase number of releases. Furthermore, the addition of new features makes the maintenance process even more complex. Culture differences and managing large team structure also reported key issues in agile environment especially in distributed settings where team dispersed in nature and more challenging task is to maintain good communication bandwidth among remote and co-located teams.

Avoidance: To avoid or minimize the communication barriers in between teams, all team members practiced a general overview discussion on product modules as well as the market view analysis from products functional point of view. That makes the more technical one's to share the logical/analytical ideas with non technical team members. In distributed team environment, most of the respondents exercising daily Scrum meetings, emailing the minutes of local Scrum meeting to the onshore teams, driving sprints demos by onshore teams and then onshore teams briefly describe to offshore teams. To avoid the slow decision making challenge, respondents made some assumptions, publish those assumptions and set a time by which decisions needed to be made. This technique was partially successful and in result stakeholders forced to interact with the team more but some delays still occurred.

Mitigation: According to the respondents, to mitigate such communication and coordination challenges, discussion schedules were managed in such a way that any time team members can coordinate with each other irrespective of that who is working on which module. This practice also becomes so handy to build issues logs related to dependent modules within the product. In slow decision making challenge, one respondent mitigate this challenge by re-planned project processes on several occasions and functionality moved from earlier iterations to later ones, giving time for decisions to be made.

4.2.2 Project/Team Management

In project or team management, respondents faced some challenges in form of project time and effort estimation, software features designing issues during iterations and team management issues. Starting with iteration designing challenges, six respondents reported that they faced some serious challenges while breaking down big features into small tasks, so that they can fit into different sprints and at the end prioritize them according to the customer needs. Another challenge faced in effort estimation, they experience that estimated duration was lesser than the effort and every deliverable seemed short of features. As the deliverable builds were not adjusted after delay in first deliverable. Moreover, Assembly line of tasks was ineffective. The dependencies should have had buffer times to avoid pushing next task forward. About team management, three respondents faced almost same challenge: they feel that many team members have a low motivation for working together. They rather commit to tasks as individuals than actively seek planning together, helping each other, or doing pair programming. They usually respect sprint backlog but this exercise happens in every sprint.

Avoidance: In response of “no careful design during iteration” challenge, most of the respondents agreed that problem cause due to unrealistic promises by the management with the client without realizing the efforts needed to develop a fully functional and quality product. This challenge can be easily avoid by giving handsome time on iterations designing with continuous involvement of the customer and then jump to the development of the features. About team management, one respondent shared his experience as a Scrum master; he had frequently tried to give suggestions of having two developers share the same task together. At the end achieved partial success but with interesting facts that lead to more improvement in the future tasks.

Mitigation: To effectively mitigate design issues during development iterations, respondents exercising by giving reasonable buffer size in task assemble line to resolve any dependency issues. To resolve time and efforts management challenges, most of the respondents reported that by arranging team meetings in start and at the end of each sprint and divide the task into small junks/sprints successfully mitigate these challenges. About team management, one Scrum master respondent shared interesting fact that we missed the greatest potential of having great minds work together. This is difficult to measure, however. In order to mitigate the lack of co-work, we have had regular (once per week or two), voluntary coding with pair programming practice used for the last 6 months. This has had a possible partial effect with the two developers occasionally sharing a task.

4.2.3 Requirements Engineering

Requirements engineering practices are playing a vital role in every system development and at the end able to finish product with complete functional requirements. Most of the today projects failed due to bad requirement engineering practices and in result faced consequences in poor quality, over budget and incomplete product functionality. In our survey, four respondents faced serious issues during their development by misunderstanding of the requirements/ stories that at the end create confusion among the team members. There is no such avoidance except by arranging startup meetings, weekly task sheet to adjust project deadlines or scope, and continuous involvement of the customer in every development phase to minimize this challenge.

4.2.4 Scaling agile practices

Scaling agile practices in whole organization is tricky especially where people feel comfortable to use their conventional processes and feel scared to adopt any change in their development cycle. Through literature, we found that this phenomenon is common in small and even in large organizations. Although, agile practices bring advantages over traditional processes, include high team productivity, low management overhead, happier customer and short sprints. However, the applicability of agile approaches/practices is constrained by number of factors in shape of experience level of project personnel, project type and size, and access to committed customer. In survey, two respondents addressed scaling agile challenge in a way that we experienced that agile methods were difficult to scale up in large and complex projects because of the lack or weak architecture planning, low level of test coverage and over focusing on early results.

Mitigation: To mitigate this challenge, they somehow used same practice: firstly, Product Managers with Architectures create the design and then the design is discussed with the team. Experience shows that giving more time in architecture designing pay at the end in term of smooth scaling of agile, less design issues that ultimately increase product quality.

4.2.5 Software Quality

Most of the respondents mentioned software quality challenges in shape of inadequate requirements verification, lack of user acceptance testing, test automation. One respondent experience that a challenge occurred whenever new functionality is written or old modified by introducing automated acceptance and unit testing into a years-long project with a large legacy code base. Writing the first unit test for a module can take several hours. Old tests break easily and most often they may not compile anymore. The full automated acceptance test run can take about 8 hours and some acceptance tests failed randomly with no good reason.

4.2.6 Technical Competency

One survey respondent mentioned the technical competency challenge in agile development. As a team, they experienced that low technical competence at the start of the project resulting in many unfinished work items during the first sprints. Missing sprint targets and abnormal burn down charts are clear indication that there's a problem in the project development. This case even worst in tight sprints where team members couldn’t able to get time to lean new ways of learning and time pressure further drop their energy. To mitigate such challenges, they replaced some of the experienced person and try to build team mixture of experience and new members to boost technical competence. This pair programming practice so far produced significant results but adaptation time was still quite significant.

4.2.7 Process Conflicts

Process conflicts are the first and most obvious area of difficulty. The problem arise when organizations experience that how they merge agile, lightweight processes with industrial standard processes without either killing agility in their development processes or undermining the years spent on defining and refining the system processes. In our survey, three respondents mentioned process conflicts in different way. One challenge is they are writing a module for a system which API is constantly being changed by another team in another company. Another challenge is corporate processes not adapted software development which causes inefficiencies and friction between the layers of the organization.

To mitigate these challenges, respondents planned their weekly teleconferences with the other teams, synchronizing their tasks with ours while iteration planning to avoid "conflicts" and relay on continuous communication bandwidth with the other teams. They also successfully mitigate process conflicts issues by exchanging compatible repositories and frequent code bundle exchanges.

4.2.8 Architecture Design

The chosen architecture by the software development team in the beginning of the development cycle may become inappropriate as development going on, new requirements become known. Architecture rework may add significant input to project cost and time schedule. Two respondents reported this challenge: due to inadequate architecture, they failed to extend or scale their architecture. To mitigate this issue, refactoring is the practice is used to altering the internal structure of the application to make it easier to modify and understand without changing its observable behavior. They rely on XP, by refactoring the design to improve and give strength to the overall architecture of the system. Experience shows that refactoring is not always obvious and depends of many factors such as schedule pressure and developers experience. Hence, refactoring is an ongoing process to improve the architecture design but often couldn’t able to fully address the inappropriate or inadequate architecture problem.

4.3 VISUALIZATION TECHNIQUES

Information visualization techniques found through industrial survey can be categorized into two main categories; Business Modeling and Project tracking and Work Status.

4.3.1 Business Modeling

Business modeling is one of the visualization area in which we had received lots of responses. Here are few techniques that are mentioned by the respondents.

4.3.1.1 Data Flow Diagrams (DFD) and Flow Charts

Data flow diagrams can be used for representation of requirements elicitation and validation. Respondents mentioned that DFDs help the developers what should be done and how should be done. It also helps to visualize the flow of data in the system. One of the respondent highlighted that visualization of data through DFDs reduces the ambiguities and reduces the development duration. Flow charts are used to clarify the requirements according to clients need.

4.3.1.2 Unified Modeling Language (UML)

Unified Modeling Language is one of the main sources of visualizing software business models; by using UML helps to visualize dependencies, and to gain agreement even with customers. Although UML is not very much admired to use in agile but it is conventional enough to answer the challenges in large-scale applications, and it is easy to be understood by different stakeholders. Seven respondents mentioned that they are using UML for modeling their requirements. They mentioned that UML helps for describing the system in logical terms which can be understood by the business stakeholders and the developers.

4.3.1.3 Use Case Diagrams

Use case diagrams are used to visualize the requirements of the system. Respondents mentioned that use case diagrams portraits the different type of users of a system and the various ways that they interact with the system. Two respondents highlighted that use case diagram helps to understand the overall design, architecture and interaction between different components. Use case diagrams help to build the understanding for management and other non-technical stakeholders.

4.3.2 Project Tracking and Work Status

Project tracking and work status is one of the other category of visualization techniques in which we had received responses from survey respondents. Here are the details of these techniques.

4.3.2.1 Project Backlog with Stories

Project backlog with stories is practiced in Scrum and Agile project management practices. Our survey respondents mentioned that project backlog is essentially a feature with sorted wish list by order of implementation. It gives us the possibility to talk easily about the different aspects of the stories, ideas and interpretations. It simplifies the task management and eases the project manager’s work.

4.3.2.2 Burn down Charts

Burn down chart counts the number of Kanban (backlog tasks) and track it in a time box to show the trend of work status. Many survey respondents mentioned that they use burn down charts because they keep the team aware about how realistic the estimations were done and how well the team is performing. Burn down charts details the comparison of work completed and work remaining effort. It is used to estimate the time for each sprint cycle completion. It helps in tracking the project and estimating if it will be done before the deadline. Potential risks of not meeting the deadline can be detected even in an early phases of the project and corrective preemptive actions can be taken when there is still not too late.

4.3.2.3 Kanban Boards

Kanban boards can be used to display the true state of development process, development speed, and clarify pending work items. One of the survey respondent said that through kanban boards the whole development process can be organized. Kanban boards are useful for developers, seeing the tasks move and get completed give a psychological boost of accomplishment to the team.

4.3.2.4 Scrum Story Boards

Six of survey respondents responded that they are using Scrum story boards as a visualization technique in their development process. One of the respondent mentioned that his team uses Scrum story board because it helps to reassign bugs when someone is stockpiling a lot of them. It also helps to identify common bottlenecks that several team members could come across off. It helps to see what other people in a team are doing so that communicate and help each other.

4.3.2.5 Gantt Charts

Gantt charts are used to illustrate the start and end dates of the secondary and primary tasks of a project. Five of our survey respondents mentioned that they are using Gantt charts to manage their tasks. Gantt charts help to schedule project tasks and allow seeing work status.

4.3.2.6 Miscellaneous Techniques

There were some miscellaneous visualization techniques, reported by survey respondents. These techniques include SA/ SD informal diagrams by using whiteboard to illustrate the business processes. By using these diagrams team can get a better understanding of the problem / solution. Better understanding of a problem means better solutions and effort estimation. One of the respondent mentioned that they use process modeling using BPMN to show the processes in details. For showing processes they use MS Visio tool to generate process models.

4.4 Context

There were 47 complete responses and one partial response, included on basis of valuable information provided for challenges of agile and information visualization techniques, received in industrial survey. There were 35 (70%) from 50 respondents described that they are using Scrum as their development approach. There were about equal number of respondents mentioned about development effort at single location and multiple locations with 25(50%) each.

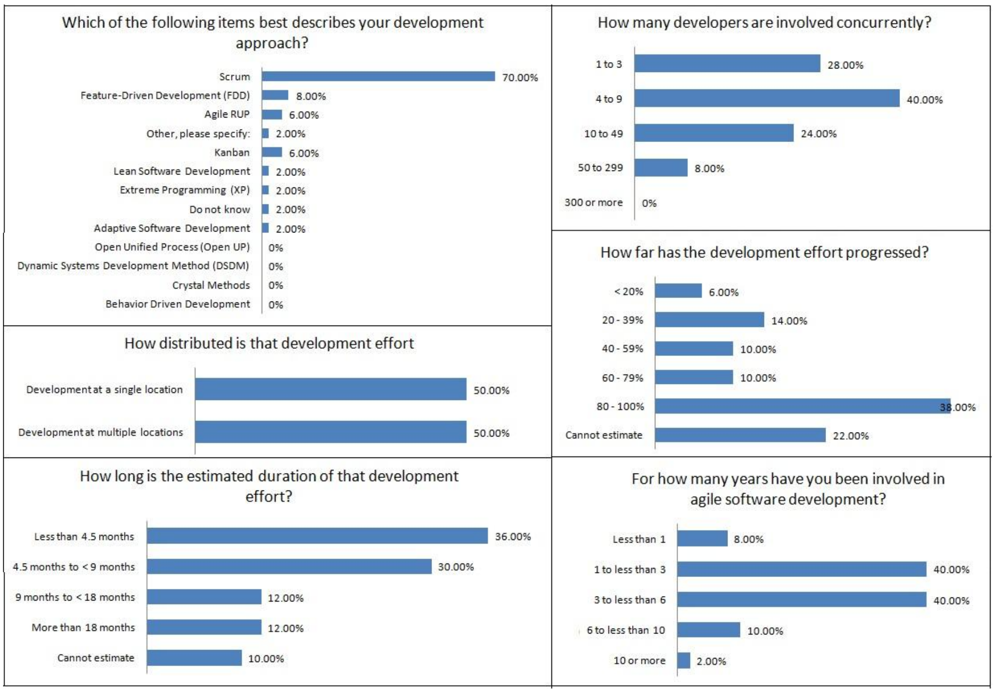


Figure 10: Summary of survey stats

4.4.1 Scenario A

By using Scrum as development approach with development at single location, estimated effort duration of 9 or less than 9 months as average duration with team size of 1 to 10 persons, 17 respondents mentioned that they face challenges like requirements engineering, project management, 3C and related to software architecture. These respondents mentioned that they are using information visualization techniques like UML and use case diagrams to visualize their business models while Scrum Task Board, Burn down charts, Backlog with stories and Scrum story board for visualizing project tracking and scheduling.

Development Approach	Scrum Development Practice
Development Environment	Development at Single Location
Estimated Effort Duration	9 or < 9 Months (Average)
Team Size	1 to 10 Persons (Average)

Table 14: Scenario A; Scrum with single development location

4.4.2 Scenario B

By using Scrum as development approach with development at multiple locations, estimated effort duration of 9 or less than 9 months as average duration with team size of 10 to 49 persons, 15 respondents mentioned that they face challenges like 3C, adoption of agile, project management, process conflicts and related to software quality. These respondents mentioned that they are using information visualization techniques like UML, case diagrams and data flow diagrams to visualize the business model while Burn down charts, Backlog with stories and physical task board for visualizing project tracking and scheduling.

Development Approach	Scrum Development Practice
Development Environment	Development at Multiple Locations
Estimated Effort Duration	9 or < 9 Months (Average)
Team Size	10 to 49 Persons(Average)

Table 15: Scenario A; Scrum with multiple development locations

4.4.3 Scenario C

By having some miscellaneous development approaches like Feature Driven Development, Agile Rup, Kanban and Mixed Approaches, 13 respondents mentioned different challenges related to requirements engineering, software quality, 3C, project management, software architectural design and some related to process conflicts. Most of the respondents are working on development at single location. These respondents mentioned that they are using different information visualization techniques to mitigate or avoid above mentioned challenges. These techniques are mainly related to business modeling like DFD, Flow Charts and UML, while some techniques related to project tracking and scheduling like Scrum task board, kanban boards and burn down charts.

Development Approach	Misc (FDD, Agile Rup, kanban, Mixed Approaches)
Development Environment	Development at Single/ Multiple Locations
Estimated Effort Duration	9 or < 9 Months (Average)
Team Size	10 to 49 Persons(Average)

Table 16 Mixed software development in agile practices

We compared responses about challenges of agile development reported by industry experts according to agile practice and development environment, shown in figure 11. We can see that if teams are using Scrum with single location then they might face challenges related to 3C, requirements engineering and project management. While 3C, software quality and adoption of agile are the challenges that might be faced when teams are using Scrum with multiple locations. Challenges related to requirements engineering, project management and software quality are mainly faced in other (FDD, Kanban, XP, etc) agile practices.

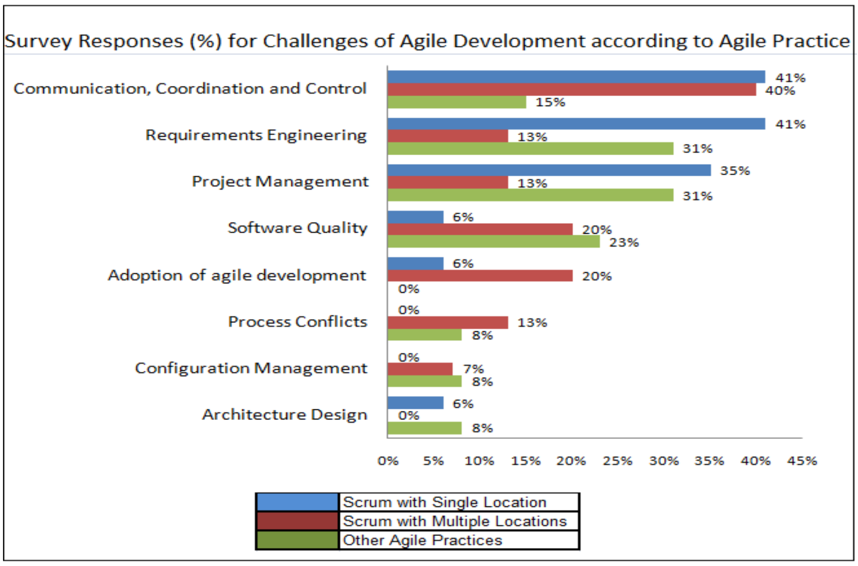


Figure 11: Challenges of Agile in large scale environment reported in survey

When we compared different information visualization techniques found through industrial survey according to agile practice with development environment, we came to results that there are few techniques that are commonly used in all situations while few produce best results in specific environment. In figure 12, we can see that Scrum story board and UML are two most reported visualization techniques by experts who use Scrum at single location. While Scrum story boards and Burn down charts are reported by experts who use Scrum with multiple locations. Data flow diagrams and Gantt charts are two most reported techniques by the experts who work in other agile methodologies.

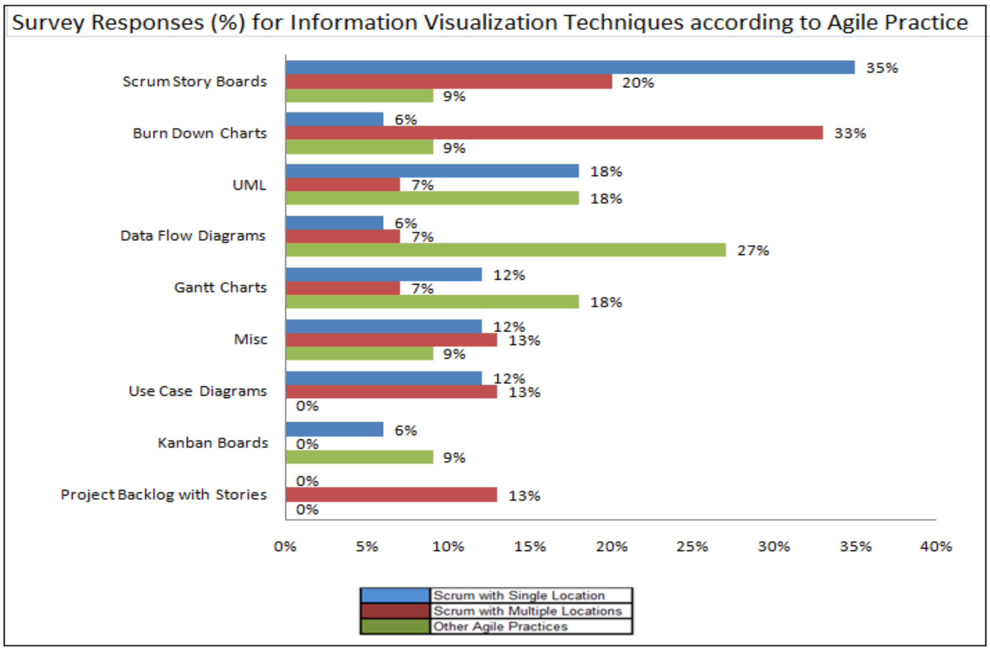


Figure 12: Information visualization techniques reported in Survey

4.4.4 Scenario D

By considering all agile practices reported in survey with respect to single development environment. We identified 33 challenges reported in agile development while 24 different visualization techniques are reported in our survey.

Development Approach	All Agile practices
Development Environment	Development at Single Location

Table 17: Survey results with respect to single location

4.4.5 Scenario E

By considering all agile practices reported in survey with respect to multiple development environments. We identified 28 challenges reported in agile development while 25 different visualization techniques are reported in our survey.

Development Approach	All Agile practices
Development Environment	Development at Multiple Locations

Table 18 Survey results with respect to multiple locations

We compared responses about challenges of agile development reported by industry experts according to development location, shown in figure 13. We can see that if teams are working single location then they might face challenges related to project management, requirements engineering and 3C. While 3C is most reported in multiple locations with project management, requirements engineering and software quality are the challenges that might be faced when teams are working at multiple locations.

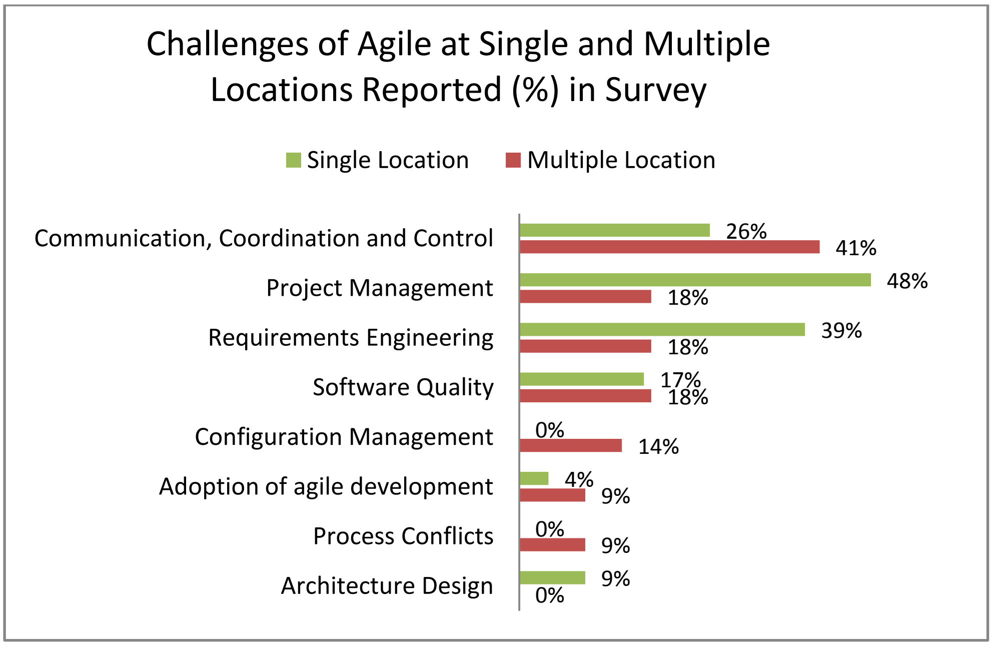


Figure 13: Challenges of agile in all development environments.

When we compared different information visualization techniques found through industrial survey according to development locations, we came to results that there are few techniques that are commonly used in all situations while few produce best results in specific environment. In figure 14, we can see that Scrum story board, Gantt charts and UML are the most reported visualization techniques by experts who work at single location. While Scrum story boards and Burn down charts are reported by experts who work at multiple locations.

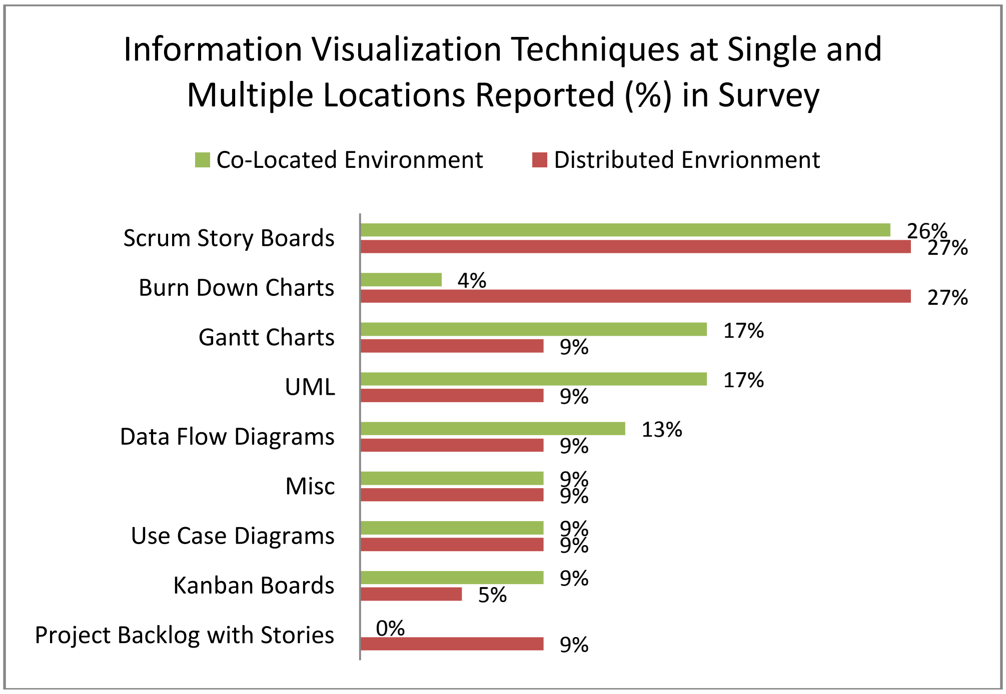


Figure 14: Information Visualization techniques in all development environments.

5 ANALYSIS

5.1 Comparative Analysis

Comparative analysis can be in two categories Qualitative and Qualitative. In qualitative analysis, entities are compared through human analytical insights while in quantitative analysis entities are compared on computational basis [120]. Comparison was performed on the lowest level of data points that are almost similar to raw data. Grounded theory [65] was applied on data extracted from literature review and survey results.

5.1.1 Challenges of Agile

There are total 84 data points for challenges of agile development were collected from literature review, these data points are grouped into 9 distinct data points. There are total 53 different data points collected through industrial survey, these data points are grouped in 10 different data points. Table 19 shows the exact number of data points found by literature review and industrial survey. There were common 75 data points from Literature Review and 43 data points from Industrial survey.

	Data Points	Common	Distinct
Literature Review	84	75	9
Industrial Survey	53	43	10

Table 19: Challenges of Agile

Through literature 9 and through survey 10 distinct categories (data points) of agile development found, after comparing both group of challenges, 8 overlapped challenges found that were reported in literature and by industrial survey. These overlapped challenges were from group of Communication, Coordination and Control, Requirements engineering, Architecture Design, Software Quality, Process Conflicts, Project Management, Configuration Management and Adoption of Agile Development. Lack of Measurement Matrices, was found only in literature review.

	Literature	Survey	Overlapped	Distinct
Challenges	9	10	8	3

Table 20: Distinct Challenges of Agile

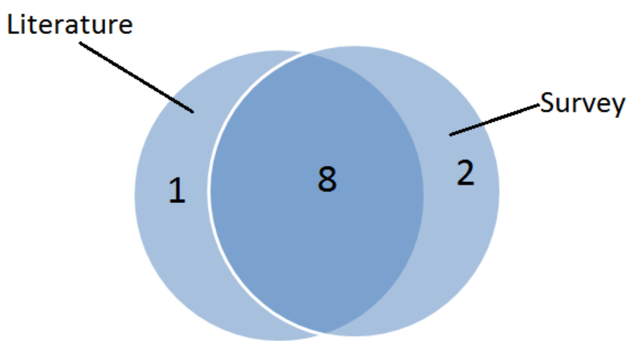


Figure 15: Common Challenges of Agile

5.1.2 Comparison of Challenges of Agile

Table 21 shows the comparison for challenges in agile development found in literature and through industrial survey. Challenges are represented by categories. In table 21 (√) shows the presence of that challenges and (X) represents the absence of that challenge.

Agile Challenges	Literature	Survey
Communication, Coordination and Control		
Communication among developers and customer	✓	✓
Cultural challenges	✓	✓
Customer access and participation	✓	✓
Design knowledge remains with the individuals	✓	✓
Collaboration between Cross-team and within team members	✓	✓
Slow Decision-making (many layers of customer and stakeholders)	✓	✓
Requirements Engineering Challenges		
Requirements engineering problems	✓	✓
Neglect of non-functional requirements	✓	X
Prioritization on a single dimension	✓	
Inadequate requirements verification	✓	✓
Misunderstanding requirements.	✓	✓
No focus on quality attributes while requirements gathering	✓	✓
Long RE duration due to complex decision processes	✓	X
Process Conflicts		
Peoples Conflicts (related to team and upper management resistance and transition issues)	✓	✓
Development process conflicts (Lifecycle, development processes, Functional n non-functional requirements)	✓	✓
Business process conflicts (organization standards, conformance of traditional process standard)	✓	✓
Adoption of Agile Development		
Integration of agile projects with existing processes	✓	✓
Automation is critical with existing processes	✓	✓
Lack of Tool Support	✓	✓
Large Number of Project Personnel	✓	X
Lack of Collaborative Office Environment	✓	✓
Area product owners	✓	X
Common sprint planning	✓	X
Scrum-of-Scrums	✓	X
Common sprint demo	✓	X
Corporate Governance (scaling Scrum to entire company)	✓	✓
Common retrospective	✓	✓
Architectural Issues		
Inadequate or inappropriate architecture	✓	✓
Architecturally very risky for new projects	✓	X
Lack of focus on architectural design	✓	✓

Configuration Management		
Minimum documentation	✓	✓
Increased maintenance effort with increase of the number of releases	✓	✓
Increase in product packaging effort	✓	X
Software Quality		
Continuous refactoring clashes existing quality control systems	✓	✓
Realize continuous testing	✓	X
Reduction of test coverage due to shortage of projects timeline	✓	✓
Lack of independent testing	✓	✓
Management Issues		
Creating and managing the requirements analysis teams	✓	X
No time for careful design during iterations	✓	✓
Flow Leveling for Limited Resources	✓	X
Team management	✓	✓
Management overhead	✓	X
Lack of Measurement Metrics		
Net Present Value Estimation	✓	X
Measuring Improvement (e.g. productivity)	✓	X
Measure Quality factors	✓	X
Difficult to judge and Map productivity on some metrics	✓	X

Table 21: Comparison of Challenges of Agile

5.1.3 Information Visualization Techniques

There are total 45 data points for information visualization techniques were collected from literature review, these data points are grouped into 21 distinct data points. There are total 47 different data points collected through industrial survey, these data points are grouped in 9 different data points. Table 22 shows the exact number of data points found by literature review and industrial survey. There were common 24 data points from Literature Review and 38 data points from Industrial survey.

	Data Points	Common	Distinct
Literature Review	45	24	21
Industrial Survey	47	38	9

Table 22: Information Visualization Techniques

Through literature 21 and through survey 9 distinct information visualization techniques (data points) found, after comparing both group of techniques, 8 overlapped techniques found that were reported in literature and by industrial survey. These techniques are Data Flow Diagrams (DFD) and Flow Charts, Unified Modeling Language (UML), Use Case Diagrams, Project Backlog with Stories, Burn down Charts, Kanban Boards and Scrum Story Boards. SA/ SD informal diagrams, visualization technique was reported only in survey while Parking lot Charts, Scheduling With A Vision (SWAV), Strawman, Cumulative Flow Diagram, 2D/3D Visualization, Mondrian, TreeMap, Sunburst, Netmap, Revision tree, User Story Mapping and Road Maps were only found in literature review.

	Literature	Survey	Overlapped	Distinct
Visualization Techniques	21	9	8	14

Table 23: Distinct Information Visualization Techniques

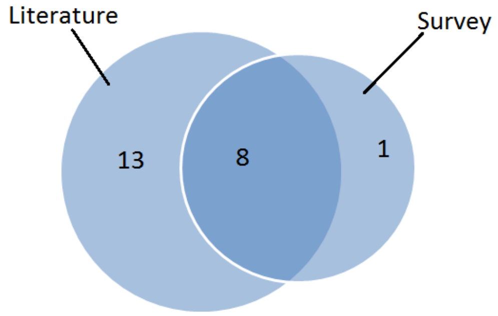


Figure 16: Common Information Visualization Techniques

5.1.4 Comparison of Information Visualization Techniques

Table 24 shows the comparison for information visualization techniques found in literature and through industrial survey. In table 24 (√) shows the presence of that technique and (X) represents the absence of that technique.

Visualization Technique	Literature	Survey
Scrum Task Board	✓	✓
Parking lot Charts	✓	X
Burn down Charts	✓	✓
Scheduling With A Vision (SWAV)	✓	X
Team Calendar	✓	✓
Strawman	✓	X
Cumulative Flow Diagram	✓	X
Data Flow Diagrams (DFD)	✓	✓
Flow Charts	✓	✓
Unified Modeling Language (UML)	✓	✓
State Charts/State Machines	✓	X
Use case diagrams	✓	✓
2D/3D Visualization	✓	X
Mondrian	✓	X
TreeMap	✓	X
Graphics Visualization	✓	X
Netmap	✓	X
Revision tree	✓	X
Scrum storyboard	✓	✓
User Story Mapping	✓	X
Road Maps	✓	X
SA/ SD Informal Diagrams	X	✓

Table 24: Comparison of Information Visualization Techniques

Information visualization techniques mentioned by industrial experts in survey can be categorized into Business Modeling and Project Tracking and Work Status. Most of the respondents reported that they use data flow diagrams (DFD), flow charts and unified modeling language (UML) for visualizing the business models. While practitioners of Scrum mentioned that they are using Scrum Storyboard, Scrum Task board and Burn down Charts for visualizing the user stories and team tasks.

5.2 Validity Threats

Some Major validity threats related to systematic literature review and survey have been addressed and briefly discussed in the following section.

5.2.1 Identification of Primary Studies

The keywords "agile" and "information/requirements visualization" are very generic terms and has been addressed in all domains like distributed and co-located settings. We found that the studies of agile development were not only published in co-located assuming small scale environment [26] but also addressing challenges in global distributed environment. In information visualization techniques, we found lot of tools and techniques that used in agile and non-agile development, we used these techniques in our proposed survey to give strength to our findings. To overcome the probability of irrelevant/missing studies, we practiced the following steps:

1) To cover the studies from different areas like economics, education and medical etc., we did not restrict our systematic literature review to software related classification code in search databases. Even though we did not finally include any article from the area related to medical, education and economics etc., but still this decision end with large number of studies in duplicates and full text search in both areas i.e. 740 and 496 in the process of findings Agile challenges and Information Visualization techniques respectively. These duplicates were due to the fact of overlapping coverage of research papers (journals) by different research databases.
2) To judge quality articles in the research study, we selected only journals publication in our primary study. This decision has further give strength by the observation that best quality of research papers presented in a conference that were afterwards published in journal as well.
3) To overcome any chance for missing research work and better coverage, we also performed manual search on gray literature [45] (e.g. Goolge scholar) for both SLRs. We followed this exercises to cross verify of our primary study results.
4) We were also keep in continues touch with one of PhD student at BTH software engineering who is working in the same discipline. This exercise was including long discussion sessions on primary study and exchange of important research material.

5.2.2 Selection of primary study and data extraction

We strictly followed the defined review protocol: search string, inclusion and exclusion criteria, and explicitly defined research databases. Further minimize the threat of publication bias, we more focused on the publications that discussed agile challenges in large scale environment along with proposed solutions and used Information visualization techniques in any phase of agile software development. However, we didn't depend much on gray literature and only used to double check our publications findings; exclude non-peer reviewed and technical reports in our review.

Further in our search findings; we used spread sheets to keep track of search results with respect to year, research methodology, development environment (distributed or co-located).

5.2.3 Survey Questionnaire

Aside from different threats associated with the systematic literature review, several threats related to the validity of questionnaire were well recognized. The instrumentation threat due to bad questionnaire design leads to the misunderstanding of the under discussion research topic [121]. We tried to minimize this threat by forming questionnaire in detailed discussion session with our supervisor and PhD student. Furthermore, we performed two weeks pilot survey on ten industrial personnel working in agile development to validate our questionnaire according to our research direction. In result, given inputs were very useful to refine our survey design and help to present questionnaire in more easy to understand and meaningful way especially in information visualization section.

The maturation threat effect by the survey participant's behavior with the sequence of time can be altering the results [121]. This usually happens when respondents acquire new knowledge during the survey filling process or become detached. We were able to deal with this threat by spreading two weeks pilot survey and add challenges and visualization techniques example links in the survey to give a short overview. Disturbance and interruption during the participation may put some influence on the results of the study [122]. However, due to the nature of online survey this threat was difficult to control by reviewers.

Validity Threat	Impact on research
Selection of Primary study	High
Data extraction from Primary Study and Industrial Survey	Medium
Survey design	High
Survey maturation threat	Low
Unavailability of Industrial Personnel	High
Industrial Personnel Biasness	Medium
Time Constraints	High

Table 25: Validity Threats

6 CONCLUSION

6.1 Discussion

In this research study, we explored applicability of different information visualization techniques for agile development available in current state of the art and currently practiced by industrial experts. We studied what are the challenges of agile development reported in literature and faced by industrial experts.

The whole research study was performed in three sequential steps; two separate systematic literature reviews were performed to find data about challenges of agile and information visualization techniques in current state of the art. In next step we validate our literature review findings through industrial experts by conducting industrial survey, grounded theory was applied on gathered data to analyze and synthesize our results. At the end, we presented answers of our research questions in the form of challenges and visualization techniques found in literature and mentioned by industrial experts. For answer of RQ3, we proposed different information visualization techniques that can be used to ease or solve the identified challenges of agile development in large scale organizations.

6.2 Answers of Research Questions

6.2.1 Research Question 1

RQ1: What are the challenges for agile development in large-scale organizations?

- ***RQ1.1****: What are the challenges for agile development in large-scale organizations available in literature?*

Challenges for agile development from literature are found through Systematic Literature Review. After applying inclusion/ exclusion criteria 26 research articles were selected. There were 84 different challenges found in primary study and by applying grounded theory we found 9 distinct categories of challenges (See Section 3.1.3). These challenges are discussed in details according to their categories in chapter 3 (See Section 3.1.4).

- ***RQ1.2:*** *What are the challenges for agile development in large-scale organizations currently faced by industry?*

Challenges for agile development faced by industry are collected through industrial experts by using online survey. There were 125 respondents, responded to our survey and 47 completed responses were gathered. There were 53 different challenges reported by industrial experts (See Section 4.2). These challenges are grouped into 10 distinct groups by applying open, axial and selective coding of grounded theory. These results of systematic literature review and industrial survey are also compared to find out the distinct challenges of agile (See Section 5.1.2).

6.2.2 Research Question 2

RQ2: Which are relevant information visualization techniques that can be used for agile development in large-scale organizations?

- ***RQ2.1:*** *Which are relevant information visualization techniques available in literature?*

Information visualization techniques that are relevant to agile development are found in literature through Systematic Literature Review. After applying inclusion/ exclusion criteria 42 research articles were selected. There were 45 different information visualization techniques found in literature (See Section 3.2.3). When we grouped these techniques there were 21 distinct technologies found. These technologies are presented and discussed in chapter 3 (See Section 3.2.4).

- ***RQ2.2:** Which are relevant information visualization techniques currently practiced by industry?*

Information visualization techniques that are relevant to agile development practices by industry are collected through industrial experts by using online survey. There were 47 different information visualization techniques reported by industrial experts (See Section 4.3). When we grouped these techniques there were 9 distinct technologies found by applying open, axial and selective coding of grounded theory. The results of systematic literature review and industrial survey are also compared to find out distinct information visualization techniques (See Section 5.1.4).

6.2.3 Research Question 3

***RQ3**: How can identified information visualization technique ease or solve the agile challenges identified in RQ1?*

By comparing and analyzing results of literature review and industrial survey we found 7 different visualization techniques that can be used to ease or solve challenges of agile development. These techniques are selected on basis of quality assessment criteria which are;

1. Technique has reported in literature and mentioned by industrial experts.
2. More than 5 industrial experts reported that technique to ease or resolve challenges.

So we can say that information visualization techniques can be used to ease, solve or mitigate challenges of agile. These techniques are found through literature review and confirmed by industrial experts.

Visualization Technique	Ease or mitigate challenge(s)
Data Flow Diagrams	Used to clarify the requirements and ease challenges related to requirements specification and flow of data in system.
UML	It helps to visualize dependencies, and to gain agreement during team meetings. Teams can use UML, if they are facing challenges related to understanding of requirements or requirements dependencies.
Use Case Diagrams	Understand the overall design and architecture, interaction between components and helps customer understanding. It can be used to minimize challenges related to understanding of developing system.
Burn Down Charts	It helps to make estimation realistic, potential risk of not meeting the deadlines can be detected even in the early phases of the projects so that corrective actions can be taken. So this technique can be used to ease or resolve challenges related to resource management, task estimation and progress tracking.
Scrum Story Board or Kanban Boards	Scrum Story Board helps to track the team progress in Scrum environment while kanban boards in Kanban environment. Both techniques can be used to communicate and help each other. If teams facing challenges related to progress tracking in Scrum environment then Scrum Story or Task Board or Kanban Board can ease or solve these challenges.
Gantt Chart	It helps to schedule project tasks and allows project managers to track project progress. It can be used to ease or mitigate challenges related to scheduling or resource tracking.

Table 26: Visualization techniques to ease of solve agile challenges

By analyzing survey responses we came to know that most of the time industrial experts are not using any standardized technique to visualize the information. In industry, experts follow semi-standardized approaches to visualize the processes and information.

6.3 Future Work

For this particular study we had identified that information visualization can be used to ease or resolve the challenges of agile development in large scale organizations. In this study we proposed 7 different information visualization techniques that can be used to ease or resolve the challenges of agile development. These techniques are proposed on the basis of their availability in literature and mentioned by industrial experts. Due to time limitation, we were not being able to validate these techniques from experts.

The work of this study can be extended by consulting industrial experts and validate these identified information visualization techniques useful in agile development. In validation process, these techniques can be validated in two angles; role of information visualization in agile and effectiveness of these techniques to ease or resolve the challenges of agile development.

7 REFERENCES

[1] S. Black, "Formal versus agile: survival of the fittest," *IEEE COMPUTER*, vol. 42, no. 9, pp. 37-45, September. 2009.

[2] A. Qumer and B. Henderson-Sellers, "An evaluation of the degree of agility in six agile methods and its applicability for method engineering," *Information and Software Technology*, vol. 50, no. 4, pp. 280–295, Mar. 2008.

[3] M. Laanti, O. Salo, and P. Abrahamsson, "Agile methods rapidly replacing traditional methods at Nokia: A survey of opinions on agile transformation," *Information and Software Technology*, vol. 53, no. 3, pp. 276–290, Mar. 2011.

[4] A. Qumer and B. Henderson-Sellers, "A framework to support the evaluation, adoption and improvement of agile methods in practice," *Journal of Systems and Software*, vol. 81, no. 11, pp. 1899–1919, Nov. 2008.

[5] M. Khalifa and J. M. Verner, "Drivers for software development method usage," *IEEE Transactions on Engineering Management*, vol. 47, no. 3, pp. 360 –369, Aug. 2000.

[6] V. Schuppan and W. Rußwurm, "A CMM-based evaluation of the V-Model 97," in *Software Process Technology*, vol. 1780, R. Conradi, Ed. Springer Berlin / Heidelberg, pp. 69–83, 2000.

[7] P. Kruchten, *The Rational Unified Process: An Introduction*. Addison-Wesley Professional, 2004.

[8] K. Schwaber and M. Beedle, *Agile software development with Scrum*. Upper Saddle River, NJ: Prentice-Hall, 2002.

[9] J. Stapleton and D. Consortium, *Dsdm: Business Focused Development*. Pearson Education, 2003.

[10] K. Beck and C. Andres, *Extreme Programming Explained: Embrace Change (2nd Edition)*. Addison-Wesley Professional, 2004.

[11] S. R. Palmer and M. Felsing, *A Practical Guide to Feature-Driven Development*, 1st ed. Pearson Education, 2001.

[12] M. Poppendieck and T. Poppendieck, *Lean Software Development: An Agile Toolkit*. Addison-Wesley Professional, 2003.

[13] I. Sommerville and G. Kotonya, *Requirements Engineering: Processes and Techniques*. New York, NY, USA: John Wiley & Sons, Inc., 1998.

[14] T. Hall, A. Rainer, N. Baddoo, and S. Beecham, "An empirical study of maintenance issues within process improvement programmes in the software industry," in *IEEE International Conference on Software Maintenance, Proceedings*, 2001, pp. 422 –430, 2001.

[15] D. Truex, R. Baskerville, and J. Travis, "Amethodical systems development: the deferred meaning of systems development methods," *Accounting, Management and Information Technologies*, vol. 10, no. 1, pp. 53–79, Jan. 2000.

[16] "Manifesto for Agile Software Development." [Online]. Available: http://agilemanifesto.org/. [Accessed: 19-Mar-2012].

[17] J. Erickson, K. Lyytinen, and K. Siau, "Agile Modeling, Agile Software Development, and Extreme Programming," *Journal of Database Management*, vol. 16, no. 4, pp. 88–100, 2005.

[18] A. Sillitti, M. Ceschi, B. Russo, and G. Succi, "Managing uncertainty in requirements: a survey in documentation-driven and agile companies," in *Software Metrics, 2005. 11th IEEE International Symposium*, pp.10 –17, 2005.

[19] M. Lindvall, D. Muthig, A. Dagnino, C. Wallin, M. Stupperich, D. Kiefer, J. May, and T. Kahkonen, "Agile software development in large organizations," *Computer*, vol. 37, no. 12, pp. 26 – 34, Dec. 2004.

[20] T. Munzner, "Process and Pitfalls in Writing Information Visualization Research Papers," in Lecture Notes in Computer Science, vol. 4950, A. Kerren, J. Stasko, J. Fekete, and C. North, Eds. Springer, pp. 134–153, 2008.

[21] P. Zhang and D. Zhu, “Information visualization in project management and scheduling,” in *Proceedings of the 4th Conference of the International Society for Decision Support Systems (ISDSS)*, vol. 97, pp. 1–9, 1997.
[22] K. Beck and C. Andres, *Extreme Programming Explained: Embrace Change (2nd Edition)*. Addison-Wesley Professional, 2004.
[23] L. Cao, K. Mohan, P. Xu, and B. Ramesh, “How Extreme Does Extreme Programming Have to Be? Adapting XP Practices to Large-Scale Projects,” presented at the HICSS “04: Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS”04) - Track 3, pp.1-10, 2004.
[24] M. Laanti, “Implementing Program Model with Agile Principles in a Large Software Development Organization,” in *Proceedings of the 2008, 32nd Annual IEEE International Computer Software and Applications Conference*, Washington, DC, USA, pp. 1383–1391, 2008.
[25] C. Hansson, Y. Dittrich, B. Gustafsson, and S. Zarnak, “How agile are industrial software development practices?,” *Journal of Systems and Software*, vol. 79, no. 9, pp. 1295–1311, Sep. 2006.
[26] T. Hildenbrand, M. Geisser, T. Kude, D. Bruch, and T. Acker, “Agile Methodologies for Distributed Collaborative Development of Enterprise Applications,” in *International Conference on Complex, Intelligent and Software Intensive Systems, 2008. CISIS*, pp.540 –545, 2008.
[27] D. J. Reifer, F. Maurer, and H. Erdogmus, “Scaling agile methods,” *Software, IEEE*, vol. 20, no. 4, pp. 12–14, 2003.
[28] A. Marchenko and P. Abrahamsson, “Scrum in a Multiproject Environment: An Ethnographically-Inspired Case Study on the Adoption Challenges,” in *Agile, AGILE ’08.Conference*, pp.15 –26, 2008.
[29] M. J. Rees, “A feasible user story tool for agile software development?,” in *Software Engineering Conference, 2002 ,Ninth Asia-Pacific*, pp. 22 – 30, *2002*.
[30] K. Petersen and C. Wohlin, “Measuring the flow in lean software development,” *Software: Practice and Experience*, vol. 41, no. 9, pp. 975–996, 2011.
[31] A. Cabri and M. Griffiths, “Earned value and agile reporting,” in *Agile Conference, 2006*, pp.6–22, 2006.
[32] M. Meyer, T. G^irba, and M. Lungu, “Mondrian: an agile information visualization framework,” in *Proceedings of the 2006 ACM symposium on Software visualization*, pp. 135–144, 2006.
[33] A. Danait, “Agile offshore techniques - a case study,” in *Agile Conference, 2005. Proceedings*, pp. 214 – 217, 2005.
[34] J. Rubart and F. Freykamp, “Supporting daily scrum meetings with change structure,” in *Proceedings of the 20th ACM conference on Hypertext and hypermedia*, New York, NY, USA, pp. 57–62, 2009.
[35] B. Paech and D. Kerkow, “Non-functional requirements engineering - quality is essential,” International Workshop on Requirements Engineering, REFSQ '04, pp. 237-250, 2004.
[36] O. C. Z. Gotel, F. T. Marchese, and S. J. Morris, “On Requirements Visualization,” in *Second International Workshop on Requirements Engineering Visualization, REV 2007*, pp. 11-17, 2007.
[37] S. Supakkul and L. Chung, “Visualizing non-functional requirements patterns,” in *2010 Fifth International Workshop on Requirements Engineering Visualization (REV)*, pp. 25 –34, 2010.
[38] N. N. Khairuddin and K. Hashim, “Requirements visualization techniques: a comparative analysis,” in *Proceedings of the 8th conference on Applied computer scince*, Stevens Point, Wisconsin, USA, pp. 391–395, 2008.
[39] S. Konrad, H. Goldsby, K. Lopez, and B. H. C. Cheng, “Visualizing Requirements in UML Models,” in *First International Workshop on Requirements Engineering Visualization, REV ’06*, pp. 1-10, 2006.

[40] C. Balan, S. Dija, and D. S. Vidyadharan, “The need to adopt agile methodology in the development of cyber forensics tools,” in *2010 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pp. 1–4, 2010.
[41] C. Maples, “Enterprise Agile Transformation: The Two-Year Wall,” in Agile, *AGILE ’09.Conference*, pp. 90–95, 2009.
[42] Christopher Lee, Luigi Guadagno, and X. Jia, “An Agile Approach to Capturing Requirements and Traceability,” CiteSeerX, 2008.
[43] N. Dulac, T. Viguier, N. Leveson, and M.-A. Storey, “On the use of visualization in formal requirements specification,” in IEEE Joint International Conference on Requirements Engineering, 2002. Proceedings, pp. 71 – 80, 2002.
[44] M. S. Feather, S. L. Cornford, J. D. Kiper, and T. Menzies, “Experiences using Visualization Techniques to Present Requirements, Risks to Them, and Options for Risk Mitigation,” in First International Workshop on Requirements Engineering Visualization. REV ’06, pp. 10–15, 2006.
[45] B. Kitchenham, “Procedures for performing systematic reviews,” Keele, UK, Keele University, vol. 33, pp. 100–105, 2004.
[46] P. Cronin, F. Ryan, and M. Coughlan, “Undertaking a literature review: a step-by-step approach,” *British Journal of Nursing*, vol. 17, no. 1, pp. 38-43, 2010.
[47] B. Kitchenham and S. Charters, “Guidelines for performing Systematic Literature Reviews in Software Engineering,” *Engineering*, vol. 2, 2007.
[48] K. Petersen and C. Wohlin, “A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case,” *Journal of Systems and Software*, vol. 82, no. 9, pp. 1479–1490, Sep. 2009.
[49] T. Dybå and T. Dingsøyr, “Empirical studies of agile software development: A systematic review,” *Information and Software Technology*, vol. 50, no. 9–10, pp. 833–859, Aug. 2008.
[50] E. Hossain, M. A. Babar, and Hye-young Paik, “Using Scrum in Global Software Development: A Systematic Literature Review,” in *Fourth IEEE International Conference on Global Software Engineering, ICGSE 2009*, pp. 175–184, 2009.
[51] J. C. de Almeida Biolchini, P. G. Mian, A. C. C. Natali, T. U. Conte, and G. H. Travassos, “Scientific research ontology to support systematic review in software engineering,” *Advanced Engineering Informatics*, vol. 21, no. 2, pp. 133–151, Apr. 2007.
[52] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, “Lessons from applying the systematic literature review process within the software engineering domain,” *Journal of Systems and Software*, vol. 80, no. 4, pp. 571–583, Apr. 2007.
[53] R. M. Groves, F. J. F. Jr, M. P. Couper, J. M. Lepkowski, and E. Singer, *Survey Methodology*. John Wiley & Sons, 2009.
[54] R. M. Czekster, P. Fernandes, A. Sales, and T. Webber, “Analytical Modeling of Software Development Teams in Globally Distributed Projects,” in *Proceedings of the 2010 5th IEEE International Conference on Global Software Engineering*, pp. 287–296, 2010.
[55] K. Kelley, B. Clark, V. Brown, and J. Sitzia, “Good practice in the conduct and reporting of survey research,” *Int J Qual Health Care*, vol. 15, no. 3, pp. 261–266, May 2003.
[56] R. K. Yin, *Case Study Research: Design and Methods*. SAGE Publications, 2002.
[57] J. C. Mitchell, “Case and situation analysis1,” *The Sociological Review*, vol. 31, no. 2, pp. 187–211, 1983.
[58] C. B. Seaman, “Qualitative methods in empirical studies of software engineering,” *IEEE Transactions on Software Engineering*, vol. 25, no. 4, pp. 557 –572, Aug. 1999.
[59] T. Gorschek, E. Tempero, and L. Angelis, “A large-scale empirical study of practitioners’ use of object-oriented concepts,” in *2010 ACM/IEEE 32nd International Conference on Software Engineering*, vol. 1, pp. 115 –124, 2010.

[60] T. Punter, M. Ciolkowski, B. Freimut, and I. John, “Conducting on-line surveys in software engineering,” in *Empirical Software Engineering, ISESE 2003. Proceedings. 2003 International Symposium on*, pp. 80 – 88, 2003.
[61] F. F. Stephan and P. J. McCarthy, *Sampling opinions: An analysis of survey procedure*. Oxford, England: John Wiley, 1958.
[62] C.E. Särndal, B. Swensson, and J. Wretman, *Model Assisted Survey Sampling*. Springer, 2003.
[63] D. I. K. Sjoberg, T. Dyba, and M. Jorgensen, “The Future of Empirical Methods in Software Engineering Research,” in *Future of Software Engineering, FOSE ’07*, pp. 358 –378, 2007.
[64] Alain Pinsonneault, École Hautes, Études Commerciales, Montréal Québec, and K. L. Kraemer, “Survey Research Methodology in Management Information Systems: An Assessment,” CiteSeerX, 1993.
[65] B. G. Glaser, *Basics of grounded theory analysis: emergence vs forcing*. Sociology Press, 1992.
[66] B. G. Glaser and A. L. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*. Transaction Publishers, 1967.
[67] J. Cusick and A. Prasad, “A Practical Management and Engineering Approach to Offshore Collaboration,” *IEEE Software*, vol. 23, no. 5, pp. 20 –29, Oct. 2006.
[68] K. Charmaz, *Constructing Grounded Theory: A Practical Guide through Qualitative Analysis*. Pine Forge Press, 2006.
[69] J. Carver, “The Use of Grounded Theory in Empirical Software Engineering,” in *Empirical Software Engineering Issues. Critical Assessment and Future Directions*, vol. 4336, V. Basili, D. Rombach, K. Schneider, B. Kitchenham, D. Pfahl, and R. Selby, Eds. Springer Berlin / Heidelberg, pp. 42–42, 2007.
[70] A. Strauss and J. Corbin, *Basics of qualitative research: Techniques and procedures for developing grounded theory (2nd ed.)*, vol. xiii. Thousand Oaks, CA, US: Sage Publications, Inc, 1998.
[71] B. Ramesh, L. Cao, and R. Baskerville, “Agile requirements engineering practices and challenges: an empirical study,” *Information Systems Journal*, vol. 20, no. 5, pp. 449–480, 2010.
[72] M. A. Babar, “An exploratory study of architectural practices and challenges in using agile software development approaches,” in *Joint Working IEEE/IFIP Conference on Software Architecture, European Conference on Software Architecture. WICSA/ECSA 2009*, pp. 81 –90, 2009.
[73] M. Paasivaara and C. Lassenius, “Could Global Software Development Benefit from Agile Methods?,” in *International Conference on Global Software Engineering, ICGSE ’06*, pp. 109 –113, 2006.
[74] P. A. Beavers, “Managing a Large ‘Agile’ Software Engineering Organization,” in *Agile Conference (AGILE), AGILE* ’07, pp. 296 –303, *2007*.
[75] M. Paasivaara, S. Durasiewicz, and C. Lassenius, “Distributed Agile Development: Using Scrum in a Large Project,” in *IEEE International Conference on Global Software Engineering, ICGSE 2008*, pp. 87 –95, 2008.
[76] D. R. Greening, “Enterprise Scrum: Scaling Scrum to the Executive Level,” in *2010 43rd Hawaii International Conference on System Sciences (HICSS)*, pp. 1 –10, 2010.
[77] E. C. Lee, “Forming to Performing: Transitioning Large-Scale Project Into Agile,” in *Agile, AGILE ’08. Conference*, pp. 106 –111, 2008.
[78] M. Ganis, D. Leip, F. Grossman, and J. Bergin, “Introducing agile development (XP) into a corporate Webmaster environment - an experience report,” in *Agile Conference, Proceedings*, pp. 145 – 152, 2005.
[79] E. Hossain, M. A. Babar, H. Paik, and J. Verner, “Risk Identification and Mitigation Processes for Using Scrum in Global Software Development: A Conceptual Framework,” in *Software Engineering Conference, APSEC ’09. Asia-Pacific*, pp. 457 –464, 2009.

[80] D. Tudor and G. A. Walter, "Using an agile approach in a large, traditional organization," in *Agile Conference*, 2006, pp.367–373, 2006.
[81] O. Ktata and G. Lévesque, "Agile development: issues and avenues requiring a substantial enhancement of the business perspective in large projects," in *Proceedings of the 2nd Canadian Conference on Computer Science and Software Engineering*, New York, NY, USA, pp. 59–66, 2009.
[82] A. Shatil, O. Hazzan, and Y. Dubinsky, "Agility in a Large-Scale System Engineering Project: A Case-Study of an Advanced Communication System Project," in *2010 IEEE International Conference on Software Science, Technology and Engineering (SWSTE)*, pp. 47 –54, 2010.
[83] A. Mahanti, "Challenges in Enterprise Adoption of Agile Methods - A Survey," *Journal of Computing and Information Technology*, vol. 14, no. 3, pp. 197–206, May 2004.
[84] H. Koehnemann and M. Coats, "Experiences Applying Agile Practices to Large Systems," in *Agile Conference, AGILE '09.*, pp. 295 –300, 2009.
[85] M. Paasivaara and C. Lassenius, "Scaling Scrum in a Large Distributed Project," in *2011 International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pp. 363 –367, 2011.
[86] P. Kettunen, "Adopting key lessons from agile manufacturing to agile software product development—A comparative study," *Technovation*, vol. 29, no. 6, pp. 408–422, Jun. 2009.
[87] E. Parnell-Klabo, "Introducing lean principles with agile practices at a Fortune 500 company," in *Agile Conference*, 2006, pp.232-242, *2006*.
[88] J. Savolain, J. Kuusela, and A. Vilavaara, "Transition to Agile Development - Rediscovery of Important Requirements Engineering Practices," in *Proceedings of the 2010 18th IEEE International Requirements Engineering Conference*, Washington, DC, USA, pp. 289–294, 2010.
[89] B. Boehm and R. Turner, "Management challenges to implementing agile processes in traditional development organizations," *IEEE Software*, vol. 22, no. 5, pp. 30 – 39, Oct. 2005.
[90] R. Castell'o, R. Mili, and I. G. Tollis, "A framework for the static and interactive visualization of statecharts," *J. Graph Algorithms Appl.*, vol. 6, no. 3, pp. 313–351, 2002.
[91] M. Glinz, "A lightweight approach to consistency of scenarios and class models," in *4th International Conference on Requirements Engineering, Proceedings*, 2000, pp. 49 –58, 2000.
[92] K. Schneider, K. Stapel, and E. Knauss, "Beyond Documents: Visualizing Informal Communication," in *Requirements Engineering Visualization, REV '08.*, pp. 31 –40, 2008.
[93] K. Wnuk, B. Regnell, and L. Karlsson, "Feature Transition Charts for Visualization of Cross-Project Scope Evolution in Large-Scale Requirements Engineering for Product Lines," in *2009 Fourth International Workshop on Requirements Engineering Visualization (REV)*, pp. 11 –20, 2009.
[94] F. Perez and P. Valderas, "Allowing End-Users to Actively Participate within the Elicitation of Pervasive System Requirements through Immediate Visualization," in *2009 Fourth International Workshop on Requirements Engineering Visualization (REV)*, pp. 31 –40, 2009.
[95] O. C. Z. Gotel, F. T. Marchese, and S. J. Morris, "On Requirements Visualization," in *Second International Workshop on Requirements Engineering Visualization, REV 2007*, p. 1-11, 2007.
[96] K. T. Hansen, "Project visualization for software," *IEEE Software*, vol. 23, no. 4, pp. 84 –92, Aug. 2006.
[97] I. Ozkaya, "Representing Requirement Relationships," in *First International Workshop on Requirements Engineering Visualization, REV '06*, pp. 3-8, 2006.

[98] J. Helming, M. Koegel, F. Schneider, M. Haeger, C. Kaminski, B. Bruegge, and B. Berenbach, “Towards a unified Requirements Modeling Language,” in *2010 Fifth International Workshop on Requirements Engineering Visualization (REV)*, pp. 53 –57, 2010.

[99] N. A. Ernst, Y. Yu, and J. Mylopoulos, “Visualizing non-functional requirements,” in *First International Workshop on Requirements Engineering Visualization, REV ’06,* pp. 25-34, 2006.

[100] B. Palyagar, P. Shanthakumar, and A. Kishore, “Visual Strawman to Relate Program RE to Project RE,” in *Requirements Engineering Visualization, REV ’08.*, pp. 1 –10, 2008.

[101] J. M. Robarts, “Practical Considerations for Distributed Agile Projects,” in *Agile, AGILE ’08. Conference*, pp. 327 –332, 2008.

[102] M. Maham, “Planning and Facilitating Release Retrospectives,” in *Agile, AGILE ’08. Conference*, pp. 176 –180, 2008.

[103] P. Heim, S. Lohmann, K. Lauenroth, and J. Ziegler, “Graph-based Visualization of Requirements Relationships,” in *Requirements Engineering Visualization, REV ’08*, pp. 51 –55, 2008.

[104] T. Merten, D. Juppner, and A. Delater, “Improved representation of traceability links in requirements engineering knowledge using Sunburst and Netmap visualizations,” in *2011 Fourth International Workshop on Managing Requirements Knowledge (MARK)*, pp. 17 –21, 2011.

[105] J. R. Cooper, S.-W. Lee, R. A. Gandhi, and O. Gotel, “Requirements Engineering Visualization: A Survey on the State-of-the-Art,” in *2009 Fourth International Workshop on Requirements Engineering Visualization (REV)*, pp. 46 –55, 2009.

[106] H. In and S. Roy, “Visualization issues for software requirements negotiation,” in *Computer Software and Applications Conference,. COMPSAC 2001. 25th Annual International*, pp. 10 –15, 2001.

[107] P. Donzelli, D. Hirschbach, and V. Basili, “Using Visualization to Understand Dependability: A Tool Support for Requirements Analysis,” in *Software Engineering Workshop, 2005. 29th Annual IEEE/NASA*, pp. 315 –324, 2005.

[108] M. S. Feather, S. L. Cornford, J. D. Kiper, and T. Menzies, “Experiences using Visualization Techniques to Present Requirements, Risks to Them, and Options for Risk Mitigation,” in *First International Workshop on Requirements Engineering Visualization, REV ’06*, pp. 10-15, 2006.

[109] H. T. Van, A. van Lamsweerde, P. Massonet, and C. Ponsard, “Goal-oriented requirements animation,” in *Requirements Engineering Conference, 2004. Proceedings. 12th IEEE International*, pp. 218 – 228, 2004.

[110] R. Ther’on, A. Gonz’alez, F. J. Garc’ia, and P. Santos, “The use of information visualization to support software configuration management,” in *Proceedings of the 11th IFIP TC 13 international conference on Human-computer interaction - Volume Part II*, Berlin, Heidelberg, pp. 317–331, 2007.

[111] C. L. Cowan, “When the VP is a Scrum Master, You Hit the Ground Running,” in *Agile Conference (AGILE),2011*, pp. 279 –283, 2011.

[112] R. A. Gandhi and S.-W. Lee, “Visual Analytics for Requirements-driven Risk Assessment,” in *Second International Workshop on Requirements Engineering Visualization, REV ’07*, pp. 6-12, 2007.

[113] M. Pichler and H. Rumetshofer, “Business Process-based Requirements Modeling and Management,” in *First International Workshop on Requirements Engineering Visualization, REV ’06*, pp. 1-6, 2006.

[114] M. Kajko-Mattsson and J. Nyfjord, “A Model of Agile Evolution and Maintenance Process,” in *42nd Hawaii International Conference on System Sciences, HICSS ’09*, pp. 1 –10, 2009.

[115] J. Dargham and R. Semaan, “A Navigational Web Requirements Validation through Animation,” in *Third International Conference on Internet and Web Applications and Services, ICIW ’08*, pp. 211 –216, 2008.

[116] S. Faily and I. Flechais, “Context-Sensitive Requirements and Risk Management with IRIS,” in *Requirements Engineering Conference, RE ’09. 17th IEEE International*, pp. 379 –380, 2009.
[117] G. Gabrysiak, H. Giese, and A. Seibel, “Interactive Visualization for Elicitation and Validationn of Requirements with Scenario-Based Prototyping,” in *2009 Fourth International Workshop on Requirements Engineering Visualization (REV)*, pp. 41 –45, 2009.
[118] H. Solheim, F. Lillehagen, S. A. Petersen, H. Jorgensen, and M. Anastasiou, “Model-driven visual requirements engineering,” in *13th IEEE International Conference on Requirements Engineering, 2005. Proceedings*, pp. 421 – 425, 2005.
[119] O. C. Z. Gotel and F. T. Marchese, “Scouting Requirements Quality Using Visual Representations,” in *Information Visualisation, 2009 13th International Conference*, pp. 519 –526, 2009.
[120] A. E. Bryman, T. F. Liao, and M. Lewis-Beck, *The SAGE Encyclopedia of Social Science Research Methods*. SAGE, 2003.
[121] C. Wohlin, P. Runeson, M. Host, C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering: an Introduction*. Kluver Academic Publishers, 2000.
[122] N. Juristo and A. M. Moreno, *Basics of Software Engineering Experimentation*, 1st ed. Springer Publishing Company, Incorporated, 2010.
[123] S. Chandra Misra and V. Kumar, “Identifying some critical changes required in adopting agile practices in traditional software development projects,” *International Journal of Quality & Reliability Management*, vol. 27, no. 4, pp. 451–474, Apr. 2010.

8 APPENDIX

8.1 Survey Questionnaire

Agile Development in Large-Scale Environment

Demographics

Agile development is well understood for small projects, but not when many people and possibly multiple locations are involved. How do ad-hoc interactions, minimal documentation, intensive customer collaboration, and continuous change work in the latter contexts? Together with you we would like to understand how to make agile development scale.

Who should complete the survey? Any practitioner involved in agile software development, including project members, managers, and other stakeholders.

What questions are asked? Questions about your development practices: your organization, your challenges, and techniques you employ.

If you would like to receive updates and survey results, please include your contact information at the end of the survey. In any case, your responses will be handled as confidential as the internet allows.

If you have any questions regarding this survey, please do not hesitate to contact us.

Thank you in advance for your participation.

1st Contact:
Indira Nurdiani, Numan Manzoor and Umar Shahzad
indira.nurdiani@bth.se, numa10@student.bth.se, umsa10@student.bth.se

2nd Contact:
Dr. Samuel A. Fricker (supervisor)
Blekinge Institute of Technology, Karlskrona, Sweden
http://www.bth.se/com/sfr.nsf/pages/samuelafricker
samuel.fricker@bth.se

About Yourself

1. For how many years have you been involved in agile software development? *

- Less than 1
- 1 to less than 3
- 3 to less than 6
- 6 to less than 10
- 10 or more

Your current or most recent development effort

We would like to know details related to the current development effort you are involved in. Development effort refers to development projects or other forms of software development, such as maintenance.

2. What is your role in that development effort? *

3. How far has the development effort progressed? *

- < 20%
- 20 - 39%
- 40 - 59%
- 60 - 79%
- 80 - 100%
- I do not know and cannot estimate

4. How many developers are involved concurrently? Indicate the maximum number. *

- 1 to 3
- 4 to 9
- 10 to 49
- 50 to 299
- 300 or more
- I do not know and cannot estimate

5. How long is the estimated duration of that development effort? *

- Less than 4.5 months
- 4.5 months to < 9 months
- 9 months to < 18 months
- More than 18 months
- I do not know and cannot estimate

6. How distributed is that development effort (moment with maximum number of developers)? *

Here distributed means, development teams are distributed across multiple locations and work on the same development effort or product.

- Development at a single location
- Development at multiple locations (at least 10% of the developers work not at the central site).

7. Which of the following items best describes your development approach? *

- Scrum
- Extreme Programming (XP)
- Dynamic Systems Development Method (DSDM)
- Feature-Driven Development (FDD)
- Adaptive Software Development
- Open Unified Process (Open UP)
- Agile RUP
- Kanban
- Lean Software Development
- Crystal Methods
- Do not know
- Behavior Driven Development
- Other, please specify: ____

8. Which of the following practices are used in your development effort?

In case you do not understand a given practice, please do not select it.

- [ ] Teams involved in full component development life-cycle responsibility (defining, building, and testing)
- [ ] Iteration planning, execution, and tracking
- [ ] Planning and tracking of small, frequent releases
- [ ] 2-level planning and progress tracking (i.e. for releases and iterations)
- [ ] Tests are written before or concurrently with the code
- [ ] Continuous integration
- [ ] Regular reflection and adaptation of process and practices
- [ ] Explicitly defined architecture to coordinate component development
- [ ] Explicitly defined vision for the software solution
- [ ] Explicitly defined roadmap for ordering and scoping the releases
- [ ] Just-in-time elaboration of requirements
- [ ] Component development with synchronized internal releases (heartbeat)
- [ ] Planned and managed communication between people and teams
- [ ] Early and continuous market releases
- [ ] Measurements and experiences are used to improve the development process
- [ ] Development performance is measured (e.g. in terms of efficiency, quality, value delivery, and flexibility)
- [ ] Other, please specify: ____________

---

9. How would you rate the success of your development effort? *

Success rate refers to achievement of any development goal, i.e., satisfying customer, delivering on time, or delivering within budget.

- ○ Definitely Successful
- ○ Likely to be Successful
- ○ Likely to be Unsuccessful
- ○ Definitely Unsuccessful

Agile Development in Large-Scale Environment

Challenges in Agile Development

10. Development is always involving challenges. What is the most important challenge you are facing when practicing agile development in your development effort?

Examples of challenges

Challenge *

What issue or problem was encountered? In which situation was it encountered? How did you and others know that it was a problem?

Avoidance

What is causing the problem? What was done to avoid the problem? Was the avoidance successful?

Mitigation

How is the problem affecting the development effort? What was done to mitigate these effects? Was mitigation successful?

Any other important challenge? *

- ( ) Yes
- (•) No

Agile Development in Large-Scale Environment

Practices and Techniques in Agile Development

12. What information visualization technique are you using in your development effort?

Information Visualization could be a UML or case diagrams, representation of processes by graphs, Gantt chart for managing schedule, or any other technique that helps you to visualize information.

Example of Visualization techniques

What kind of visualization technique are you using? *

Visualization technique [ ]

Purpose of technique use [ ]

Tool for visualization (optional) [ ]

How does this technique help to overcome or reduce your stated challenges in agile development ?

Do you want to add another visualization technique? *

- ○ Yes
- ◉ No

Agile Development in Large-Scale Environment

Closure

If you would like to be informed about the progress and results of this research, please fill in the following information.

13. Name (optional)

14. Email (optional)

15. What else would you like to share with us? (optional)

Thank you for participating in this survey.