A Review of Requirement Engineering Issues and Challenges in Various Software Development Methods

Tejas Shah
M.Sc. (I.T.) Programme
Veer Narmad South Gujarat University
Surat, India

S V Patel
Department of Computer Science
Veer Narmad South Gujarat University
Surat, India

ABSTRACT
The Requirement Engineering (RE) is a systemic and integrated process of eliciting, elaborating, negotiating, prioritizing, specifying, validating and managing the requirements of a system. The detailed and agreed requirements are documented and specified to serve as the basis for further system development activities. The software industry has moved from traditional software development method to service oriented software development. While many researchers and practitioners have observed issues and challenges in Requirement Engineering phase specific to a software method, very little attention has been given to investigate diversity of issues and challenges of RE in different software development methods under one umbrella. This paper tries to review significant issues and challenges of RE from traditional software development method to recent service oriented software development method. The study unveils that there is a wide scope for developing new approaches and techniques in requirement engineering to resolve problems observed in various SE methods. The review discussion reveals the need of standardization and automation of RE process especially for Service oriented software development.

Keywords
Requirement Engineering, RE phase, SE methods, traditional software development, service oriented software development, automation of RE

1. INTRODUCTION
Requirement engineering (RE) is the process by which the requirements of the systems are determined. RE involves the activities of discovering the needs of stakeholders, understanding the context of requirements, modelling, negotiating, validating, documenting and managing these requirements. The factors and trends like potential increase in the scale of software systems, tighter integration of software and its environment, greater autonomy of software to adapt to its environment, and increasing globalization of software development makes the RE phase more challenging.

The organization can achieve significant business benefits by preventing problems as early as in the requirements engineering (RE) phase instead of waiting until the project finished [1]. The RE discipline is known to be crucial for the success of every project. Hall et al. in [2] reports that a large proportion (48%) of development problems stem from problems with the requirements. Moreover, fixing requirements-related problems consumes a high cost of rework in later states [3] [4]. The chaos report from the Standish Group [5] states that 44 % of the reasons for failed projects have their origin in insufficient RE.

This review aims at observing challenges and issues present in different software development manifestos. If these problems are not addressed carefully, it might hinder the adoption of the method successfully and may have negative consequences like missed schedule and overrun budget. While many researchers and practitioners have observed issues and challenges in Requirement Engineering phase specific to a software method, very little attention has been given to investigate diversity of issues and challenges of RE in different software development methods under one umbrella. The study unveils that there is a wide scope for developing new approaches and techniques in requirement engineering to resolve problems observed in various SE methods. The paper is organized as follows; Section II elaborates common issues and challenges of RE applicable to almost all software development methods. Section III to VII covers problems of RE in respective software development method. Section VIII shows summary and analysis of the review and section IX gives conclusion and future directions.

2. COMMON ISSUES AND CHALLENGES OF RE IN SOFTWARE DEVELOPMENT METHODS
For software development many methods exist and it is not feasible to include all of them here hence we have chosen widely used methods for studies. These methods include Traditional software development, Object oriented software development, agile software development, Component based software development (CBSD) and Service oriented software development (SOSD). We first analyze common issues and challenges which are found in almost all software development methods.

2.1 Realization of Security at RE Level:
As the recent systems becomes more pervasive, mobile and operational by many users, the critical processes and data has been the target by security attacks [6]. The efforts have been made to identify, model and protect threats and vulnerabilities in [7] [8] [9] [10] [11]. This approach to RE is reactive and focuses on low level security requirements. The work on high level security policies on methodologies for structuring, modeling and reasoning is done in [12] [13] [14]. But the behavioral specification of threats, attacks should be optimized at RE and design level. So to realize the security and privacy degree at RE level is a prominent challenge related to Requirement Specification.

2.2 Integration of RE Models:
The modeling conventions, methodologies and strategies simplifies the RE techniques. Modeling theory which incorporates RE modeling elements is described in [15]. Most research projects focuses on a single RE problem such as elicitation and there has been little work on interconnection of

requirement models and combining RE phases. Well defined approaches are required to interrelate RE goals, scenarios, data, functions, state-based behavior and constraints. This problem affects almost all RE phases and subsequent effect will be on architecture and design of the system.

2.3 Elicitation Technique Selection:

There are plenty of elicitation techniques available in the industry for completing elicitation tasks. All the techniques are used in hybrid manner to gather requirements from customers. But, there is no standardized technique dedicated for the respective paradigm.

2.4 Requirements Reuse:

The reuse of existing requirement artifacts makes the RE task more prescriptive and systematic [6]. The reusability of requirements facilitates the advantages at design level as well as in the development of related domain system or applications. A key challenge is to identify maximum number of reusable requirements for particular domain and how to map and model them [16].

2.5 Improvement in Requirement Quality:

The requirements elicited from stakeholders may be ambiguous, incomplete, inconsistent, incorrect and out-of-date. Some requirements are specified with only technical jargons rather than business domain terminology. The Quality Assurance (QA) task should be initiated from the RE phases itself and continues towards subsequent phases. To improve the quality of the requirement is a big challenge for all the software development methods. Because poor requirement quality heavily increases development and sustainment cost and results in delayed schedule [17].

To maintain the quality of the requirement, the inspection should be used to verify and ensure that all of the requirements have appropriate characteristics. The requirement engineers, stakeholder and evaluators need to be properly trained and required to collaborate with each other to rework on requirement until requirements turn out to be feasible and verifiable [17].

2.6 Missing Requirements:

The customer is not aware of giving 100% of what he wants for the system to be developed. The mid-size and large system ends up with thousand of requirements and derived into many subsystems. It's very hard to spot some missing requirements and their absence is often missed until the system is integrated, tested or deployed [17].

The requirement engineer must actively elicit the requirements from all the group of stakeholders. Mature methods and techniques can be used to deal with the challenge of missing requirements.

2.7 Semi-Automatic Process for Generic Template Creation:

Currently, there are many templates available for collecting requirements in an interactive manner. These templates are collecting the requirements using word documents or excel sheet. There are some RE tools available for requirement traceability and management. However, these tools and templates are not having structured process of mapping requirements to design. This is a big challenge for moving towards the process of semi-automatic requirement engineering.

2.8 Excessive Requirements Volatility:

The use of iterative, incremental model of software development is motivated because of adapting continuous changing requirement. But, if requirements changes in uncontrolled manner, then it may have substantial effect on existing architecture and design. The too much volatility may change the scope of the system too [17].

To manage the change, the requirements must be baseline and frozen at appropriate milestone for each release of the system. When there is nontrivial change of the requirements, budget and schedule requires modification [17]. To implement this, we have to take the effective steps like limiting the number of changes, scale of change within the bounded scope.

2.9 Inadequate Requirements Management:

Many projects store and manage their requirements in paper documents, spreadsheets with disparate formats managed by different profile teams. The decentralized and individual management of scattered requirements makes it difficult to authorize team members for performing operations on the requirements [17]. In case of global software development, dynamic changes take place at all the sites and management of distributed requirements is hard to implement [18].

The metadata of requirements like status, priority, rationale should be linked and stored in a compatible tool with authorization feature [17].

2.10 Accuracy and Performance Measurement of Requirements:

The performance engineering of the software checks the overall performance of the product developed. The business requires new value creation and that is reason of emergent need of performance engineering in software. But when requirements are incepted and specified, negotiation and prioritization parts decides which requirements are finally crucial and to be included in system development. However the accuracy and performance measurement matrices are required for RE activities.

2.11 Interactive RE Tool Support:

Many companies are using specification document, simple spreadsheets or RDBMS tables to store and manage their requirement. The requirements and their associated models, diagrams are often developed and stored in different incompatible tools which don't provide proper traceability of requirements [17]. Without adequate, compatible and integrated tool support; requirements becomes inconsistent, bulky, untraceable and out-of-date.

The compatible, versatile, powerful and user friendly tool should be used to capture requirements and their diagrams, associated text and metadata. The tool should have features like efficient elicitation interface, traceability wizard, negotiation layout and integration of central repository.

2.12 Communication Gap:

Irrespective of software method, the communication gap between customer and RE team is a major problem. This gap carries the problems towards modeling, design and implementation [19]. The distance gap in global software engineering complicates this gap even though synchronous and asynchronous tools are used. The stakeholders' language and culture diversity increases the complexity in requirement elicitation and negotiation in global software development. It's difficult for the development team to analyze and remove the inconsistencies, conflicts and redundancies when the

customer uses diverse nomenclature in specifying the requirements [18] [19]. This requires need of a traceability tool to monitor the communication activities in an efficient manner.

2.13 Conflicting and Ambiguous Requirements:

The different stakeholders' opinions, objectives, needs may have different meanings and may conflicts with vague words. When eliciting the requirements; the terminology, keywords and domain knowledge should be properly notified [20]. The methods should be well described to resolve the conflicts of requirements.

To remove the ambiguity and conflicts, the collected requirements can be stored in graphics prototype with the proper techniques. The customers can check this prototype and remove any conflicting requirement in step-wise refinement model [20].

2.14 Elimination of Irrelevant Requirements:

The set of requirements inquired from stakeholders may include some points which are not at all necessary. Major defects can be encountered if bad and irrelevant requirements are elicited. The process of eliminating unnecessary requirements is time consuming. The step-wise refinement model solves the issue of irrelevant requirements with the evaluation of graphics prototype by the customer [20].

2.15 Prioritization of Requirements:

The stakeholder's wavering mindset changes the priority of the requirements. Eliciting requirements from stakeholders by their position in the organization complicates the process of priority assignment. The identification strategy is required to give rating on priority requirements.

3. ISSUES AND CHALLENGES of RE in TRADITIONAL SOFTWARE DEVELOPMENT

Traditional methodologies are characterized by a sequential series of steps like requirement definition, planning, building, testing and deployment. First, the client requirements are carefully documented to the fullest extent. Then, the general architecture of the software is visualized and the actual coding commences followed by various types of testing and the final deployment [21].

The traditional software development manifesto requires the user to provide a detailed idea of the exact requirements with respect to the intended software. This methodology have a well-defined requirements model which works as a reference to implementation and coding process for the development team. The development team will perform the coding according to the documentation provided by the business analysts until the system is complete and only then it will be presented to the clients as final product [22].

The following are the issues and challenges of the RE phase abided by the traditional software development ideology.

3.1 Addressing NFR (Non Functional Requirement):

The NFR includes the indirect attributes of the system like security, privacy, portability, scalability, quality, operability and many more. The traditional software development method generally gives less attention to NFRs. For including NFRs into system the NFR repository can be created which stores NFR for different domains. We can have the interactive interface to collect NFR attributes from the user in selective mode.

3.2 Poor Requirement Traceability:

The tracing of the requirement is mandatory task to link the source of requirements to the design phase. In many projects, requirement tracing is manual process and mapping of requirement to design and architecture is difficult even with the modern tools used. The poor requirement traceability makes it difficult to accommodate proposed and actual changes [17]. The requirement traceability matrix and tool is needed to monitor the activities of requirements.

3.3 Immutable Requirements:

In waterfall model, requirements are frozen when all the stakeholders are agreed upon with what system is to be developed. However there is a tendency of the customer to change the requirement at any time during the development. There will not be any movement in requirements once specification phase gets over. This problem is solvable by using iterative software engineering models.

3.4 Elicitation End Point:

Depending on the paradigm used, the elicitation process will stop or continue even after product delivery. The developers are facing complexity in deciding the end points for elicitation in traditional software development. This problem can be easily recognized if all the stakeholders are agreed upon freezing the requirement elicitation process.

3.5 Business Agility:

As per the market demands, competition and the behaviour of the system; the business process of the organization must change to gain the strategic advantage. These changes of business requirement can occur at any time. The development methodology of traditional development remains same and becomes obsolete and outdated with no inclusion of emergent business processes [23]. The agile methodology solves this problem with the support of agility and many other features.

3.6 Customer Involvement:

In traditional approach, the customer will give their all needs and requirements only in the elicitation phase. But there will be possibility of vast amount of requirements discovery by the stakeholders at the lateral stages. Because involvement of customer is limited to elicitation and specification phase, creative and other functionalities of the system acquired from the customers will not to be accommodated in intermediate phases of the development. Moreover, the less involvement of customers creates problems in negotiation and validation of the requirements.

4. ISSUES and CHALLENEGES of RE in OBJECT ORIENTED SOFTWARE DEVELOPMENT

The industry has already passed through the major method change as object oriented development after the traditional software development for the large scale information systems. Object oriented requirement engineering is an approach to encapsulating information about process and product, as well as functionality into requirement objects [24].

In object oriented software development method requirements are directly represented as requirements objects which can be organized in generalization hierarchies that reflect different

kinds of requirements [25]. This section presents some of the issues and challenges faced in the object oriented software development manifesto.

4.1 Functional Requirement Modeling:

In object oriented software development, UML based RE approach develops set of use cases for the particular scenario of the problem. But use cases are not object oriented and doesn't specify the functional requirements of the system. In addition, representation of requirements using class may blur the concepts of domain objects [26]. The problem of modeling and specifying the domain object remains challenging problem for the researchers.

4.2 User Centric Requirement Analysis:

The requirement analysis phase should pay attention towards requirements from user perspective. The model oriented requirement engineering (MORE) framework addresses the requirement in natural language and focuses on document specific requirements [27]. The analysis of requirement is not based on user role and semantic identification of objects. The requirement objects should be represented as semantic notations which can be understood by the user [28].

However, the business objects, their relationships and detailed analysis of semantic objects are represented with user centric approach for large scale information system [28].

4.3 Poor Emphasis of NFR in Use Case Modeling:

The use cases are not representing the NFRs properly. The challenge is to include exceptional conditions and path to enable reliability, security, availability and other non functional requirements in use case modelling [17].

The parametric evaluation and analysis of security measures are elaborated with many approaches to address the security for RE [29]. The misuse cases approach is the inverse of the standard use cases and describes functions that the system should not allow [30]. These solutions are prevalent but not explicitly addressed in use case design.

5. ISSUES and CHALLENGES of RE in AGILE SOFTWARE DEVELOPMENT

In agile methodology the development team is working in small iterations and deliver portion of working software at the end of iteration. This methodology emphasizes more on customers' involvement in the development process. The agile methodology post-dates the traditional one in the evolution of the software development processes and less rigorous. Agile developers recognize that software is not a large block of a structure, but an incredibly organic entity with complex moving parts interacting with each other [21].

When agile development teams are distributed geographically in onshore and offshore location, lack of communication is major challenge for requirement elicitation. Agile method provides the freedom of making changes in the requirements even late during software development. The change in the requirements during early iterations removes the ambiguities and minimizes the chance of implementing those requirements later in the software which is very costly [31].

This section describes the issues and challenges of requirement engineering pertaining to the agile based software development method.

5.1 Conflicting Viewpoints amongst Team:

The agile team members must use the same technical language in understanding the requirements. When agile team is distributed in off-shore locations having larger time differences, it becomes difficult to have efficient coordination between team members. There may be conflicting viewpoints amongst team members which affects the particular iteration to be delivered to the customer [32].

5.2 Schedule Variations:

In the view of changes in the requirement and nature of agile methodology, project schedules vary heavily in this methodology. To control the project completion schedule, proper training of customers to specify there requirements is necessary.

5.3 Lack of Standardized RE Activities:

There are no documented RE activities which can be followed to obtain the user requirement in efficient manner [33]. Agile releases are too frequent and emergent requirements are incepted from the customers in every release. The agile manifesto and all the methodologies should have standardized and documented set of RE activities.

5.4 Incompatible Interface:

The agile product delivery turns out to be partial releases. The product developed in one phase might not be compatible with the next phase. The evolving requirements of the customer and previous released version can have semantic gaps in the features [33].

5.5 Difficulties in Evaluating NFR:

There is no specific approach for incorporating and evaluating the NFR (Non functional requirement) in agile software development. The partial release is delivered to the client and he provides feedback for functional and non functional requirement. But, the client is not having sufficient time to evaluate the quality criteria of each release of the product [34].

6. ISSUES and CHALLENGES of RE in COMPONENT BASED SOFTWARE DEVELOPMENT (CBSD)

The component is a non-trivial, independent and replaceable part of a system that fulfils a clear function in the context of a well defined architecture [35]. The CBSD method includes purchasing or constructing components. The development from already build COTS (Commercial off the Shelf) reusable components provides shorter development time and reduced cost benefits. The requirement engineering activities for acquiring, modeling and managing stakeholder needs are somewhat different in CBSD system. The issues and challenges of CBSD are as follows.

6.1 Requirements Instability:

The market of COTS products is volatile and changes rapidly. New components for a specific application domain are delivered continuously in the market. The customer evaluates the current version of component and may update the requirement specifications. This instability may affect the next version of the component and continuous change in requirement specifications [26].

6.2 Non-triviality in Selection Process of COTS Components:

There is no standardized process of selecting COTS components as per the requirement. Generally, most organizations perform the process of selecting COTS components in an ad-hoc manner. The evaluation criteria for selecting COTS components are subjective and ambiguous. So customer needs are not effectively described and the process turns out to be non-trivial [26].

6.3 Evolving Requirements during Development:

When the evaluation process of selected components is in progress, the new COTS product version may be released with added functionalities for the same domain. Another issue is of new requirement discovery. At the moment when system is integrating the component, some requirements will be known after the initial evaluation [26].

6.4 Additional Constraints Specification:

The COTS components require wrapper and glue which isolate unwanted functionalities and provide functionality to integrate different components. Generally, the COTS components are not interoperable with some systems. The system will include additional constraints due to mismatch of components with system's architecture. The connectors are not reliable and taken from 3rd party during the construction process. This will give negative contribution to system's overall quality attributes and one has to rely on included components [26]. At the time of specification, additional desired constraints have to be incorporated to assist the developer to adapt and tailor COTS components.

7. ISSUES and CHALLENGES of RE in SERVICE ORIENTED SOFTWARE DEVELOPMENT (SOSD)

Service-Oriented Computing (SOC) is emerging as the most prominent and promising software development method to deal with the constantly increasing information and software system complexity. This method is increasingly adopted by public and private organizations and its introduction makes 'Software as a Service" a unique possibility.

The concept of web service [36] enables the creation of new business models with the help of Service Oriented Architecture (SOA) which provides the environment for distributed, modular and collaborative software development [37][38]. The SOSD is using various service definition and access standards like WSDL (Web Services Description Language) [39] and SOAP (Simple Object Access Protocol) [40].

The observed issues and challenges of RE in SOSD specifically related to service specification, discovery, including NFR in service description are described in the next section.

7.1 Refinement of Specifications after Service Discovery:

Service discovery is the important phase which deals with locating correct service according to user requirements [41]. The discovery can be easily done with the help of UDDI, but SOSD should comprise of automated dynamic service discovery with high level language support.

To improve the completeness of requirement specifications, iterative discovery process is required [42][43][44]. From the consumer's side, specification refinement progress leads towards iterative discovery based on changed specifications. As the numbers of services are increasing continuously over the internet, it is a challenging task to find and select appropriate service amongst all available automatically.

7.2 Innovation and Creativity in RE:

Requirements engineering is a creative process in which stakeholders and designers work together to create ideas for new systems that are eventually expressed as requirements. The innovation provides insights in developer's mind to apply innovative ideas in creating desired properties of the future system [45].

The SOSD requires new ideas, innovations in discovering, managing and giving required service to consumers. The service provider has to quickly update the service according to the market demand and competition. The service provider can conduct brainstorming techniques and RAD/JAD workshops to make tangential reference for creative thinking in service development. [45][46].

7.3 Customer Acceptance on Service Change:

The consumer's requirements appear progressively while using service in practice. To manage the evolved services is a challenging task because shallow changes are localized but deep changes may have cascading effect on the other enterprise services or services of the business partner [47].

The SOSD method should connect RE and design phase for redesign and redeployment of the services when consumer's specifications will change and evolve over time. When services are brought from hybrid environments, there may be semantic gaps which should be taken care of in RE and design part as well [48]. Before mapping web service towards design, customer acceptance is highly preferable.

7.4 Clustering of Services as per Requirement:

Services can be grouped according to their domain and area. Clustering process of service is desirable and it puts services in the respective category. This process will reduce the searching and discovery time and increases the domain knowledge of the stakeholder. The SOSD requires a good knowledge management strategy for clustering of services [49].

7.5 Identifying Business Process at RE Level:

Web services are essential assets of the organization if properly composed with business requirements. Identification process of services by their business goals and intentions is a challenging task. The web service is playing an important role in inter and/or intra organization business process management [50]. Business processes of the organization can be easily converted into services.

The Services are designed to automate business requirements. Most of the current service description methods fail in describing business processes in detail [51]. This will raise the problem of business-software realization [52] and developed services are not conforming to the business requirements.

7.6 Lack of NFR Description:

The syntax based technique like WSDL describes services in terms of various operations. But WSDL fails in including and describing NFR [53] [54]. Moreover, SOA projects are based

on business processes and their transformation is based on composition of services and not on use cases.

7.7 Changes at SLA Level:

Changes in already operational services may require the adaptation in Service level agreement (SLA) also. Once services are in operation, handling changes at the SLA level is difficult to implement because one has to involve customer and change the rules, policies and protocols for using and composing the services [53][55].

8. SUMMARY AND ANALYSIS OF THE REVIEW

Every software development method is having its own strength and weakness and developed with evolving market needs; however some of the issues and challenges affect further stages of all software methods. After the analysis and observation of the review, an applicability table (table 1) is created which maps the challenges and issues found and present in different software development methods. Issues and challenges are categorized as per their applicability with different methods. The comment in the cell shows the reason for its inclusion. The symbols of the applicability are: √ - Applicable, ⌐ - Partially Applicable, ≠ - Not Applicable

Table1. Applicability of Issues and Challenges to Software Development Methods

No.	Issues and Challenges	Traditional Software Development	Object Oriented Software development	Agile Software development	CBSD	SOSD
1	Improvement in Requirement Quality	√	√	√	√	√
2	Inadequate Requirement Management	√	√	⌐ Frequent short releases	⌐ Managed through component repository	⌐ Managed through UDDI
3	Lack of Standardized RE Activities	√	√	√	√	√
4	Communication Gap	√ Formal Communication	√	√ Informal Communication	√	⌐ Exists till service discovery
5	Integration of RE Models	√	√	√	√	√
6	Business Agility	√	√	≠	≠	≠ Considered in service specification
7	Innovation and Creativity in RE	√	√	√ Creativity in agility and in release	√	√ Creativity in developing services
8	Elicitation Technique Selection	√	√	√	√	√
9	Requirement Reuse	≠ Static artifacts	√	√	√	√
10	Conflicting Viewpoints Amongst Teams (Global S/W engineering)	√	√	√	√	√
11	Missing Requirements	√	√	⌐ Discovered in next iteration	⌐ Stated at the time of updates	⌐ Discovered when Refining Service Specification
12	Excessive Requirements Volatility	≠	√	√	⌐ Exists till component configuration	⌐ Exists till service composition

13	Accuracy and Performance Measurement of Requirements	√	√	√	√	√
14	Semi-Automatic Process of Generic Template Creation	√	√	√	√	√
15	Interactive RE Tool Support	√	√	√	√	√
16	Realization of Security at RE Level	√	√	√	√	√
17	Conflicting and Ambiguous Requirements	√	√	⌐ Can be clarified in next iteration	⌐ Component repository removes ambiguity	⌐ UDDI manages service conflicts
18	Elimination of Irrelevant Requirements	√	√	⌐ Eliminated in next iteration	≠	≠
19	Prioritization of Requirements	√	√	√	√	√
20	Schedule Variations	≠	≠	√	≠	≠
21	Customer Involvement	√ Low Involvement	√	≠ Continuous customer interaction	⌐ Involved till searching	⌐ Involved till discovery
22	Incompatible Interface	≠	≠	√	≠	≠
23	Poor Requirement Traceability	√	√	≠	≠	≠
24	Immutable requirements	√	√	≠ Late changes can be adapted	≠	≠
25	Elicitation End Point	√	≠	≠	≠	≠
26	Functional Requirement Modeling	⌐ Few modeling techniques	√ Difficult to model domain object	≠	≠	≠
27	User Centric Requirement Analysis	√	√	≠	≠	≠
28	Poor emphasis of NFR in Requirement Modeling	≠ Less use of use case modeling	√ Security privacy patterns in use cases	⌐ Handled at agile modeling	⌐ Considered in component specification	⌐ Considered in service specification
29	Requirements Instability	≠ Stable once specified	≠	⌐	√ Volatility in COTS products	⌐
30	Additional Constraints Specification	≠	≠	≠	√ Mismatch of components	≠ Universal web service
31	Non-triviality in Elicitation	≠	≠	≠	√ In component selection	√ In service selection
32	Evolving Requirement during Development	≠ Requirements are frizzed	≠	√	√ New specification after evaluation	√ Changes in specification after orchestration
33	Refinement of Specification after Service Discovery	≠	≠	≠	√ Discovery of components	√ Discovery of web service
34	Consumer Acceptance on Service Change	≠	⌐ Reuse of requirements	⌐ Emergent changes accommodated	√	√ Changes in consumer specification
35	Clustering of Services as per Requirement	≠	≠	≠	√	√

36	Identifying Business Process at RE Level	≠	⌐ Business object identification	√	⌐ Business component mapping	√ Mapping web service in intra/ inter organization
37	Lack of NFR Description	√ NFR specification is missed	√	√	√	√ Service description methods not includes NFR
38	Requirement Change Management	⌐ Changes in Agreement	⌐ Changes in Agreement	⌐ Changes in Agreement	√ Change rules of components configuration	√ SLA change
√ **Applicable**		⌐ **Partially Applicable**		≠ **Not Applicable**		

There are 38 issues and challenges covered pertaining to various software development methods. The figure 1 indicates the distribution showing applicability of issues and challenges in different methods. It is observed that agile software development and SOSD methods are having less applicable issues and challenges as compared to the other methods.

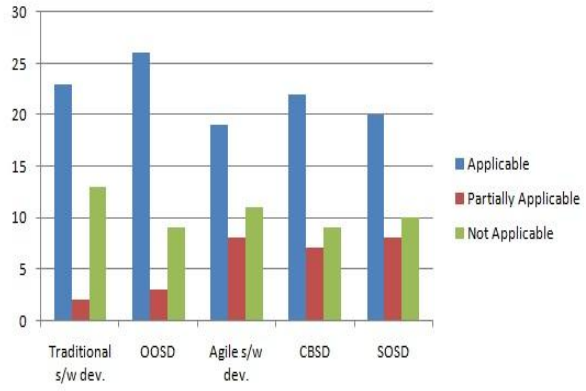

Figure 1 Distribution for applicability of issues and challenges

The table 1 describes the issues and challenges in 3 different levels. The industries have come across with many significant issues and challenges which are observed in all the software development methods. The surfaced and filtered issues and challenges are listed below in table 2 showing 10 points out of 38 issues and challenges which may have rigorous effect in all the software development methods. All these points should be resolved and handled properly at all the levels of software development.

Table 2 Significant Issues and Challenges

No.	Issues and Challenges
1	Improvement in Requirement Quality
2	Lack of Standardized RE Activities
3	Integration of RE Models
4	Elicitation Technique Selection
5	Conflicting Viewpoints Amongst Teams (Global S/W engineering)
6	Accuracy and Performance Measurement of Requirements
7	Semi-Automatic Process of Generic Template Creation
8	Interactive RE Tool Support
9	Realization of Security at RE Level
10	Prioritization of Requirements

9. CONCLUSION AND FUTURE DIRECTIONS

This review is an attempt to study issues and challenges of RE in major software development methods. In particular, this review revealed five methods and their RE problems observed in the industry. The observed key points are: improvement in requirement quality, realization of NFR at RE level, elicitation technique selection, communication gap with customers, poor requirements traceability, RE tool support, prioritization of requirements, requirement change management. Although the industrial practices have resolved some issues and facilitate some solutions to overcome it, the significant challenges still remain unattended.

These problems give insights to RE practitioner and researcher to produce high quality of software in terms of a final product, agile partial release, a COTS component or a web service. While we cannot generalize from one review, further research is needed to explore the unidentified challenges in all the software methods and developing methodologies or techniques to resolve them at the RE state itself. Moreover, the recent challenges faced of RE in SOSD like creativity in RE, refinement of service specification, inclusion of NFR in service description languages and clustering of services need attention of RE researchers for developing automated/semi-automated RE framework for SOSD.

10. REFERENCES

[1] Sommerville I, Software engineering. 7thEdition. Addison-Wesley, Harlow, 2004

[2] Hall T, Beecham S, Rainer A, 2002, "Requirements problems in twelve software companies: an empirical analysis", Software IEEE Proceeding 149(5):153–160

[3] Boehm B, Papaccio P, 1988, "Understanding and controlling software costs", IEEE Trans Software Eng 14(10):1462–1477

[4] Leffingwell D, "Calculating your return on investment from more effective requirements management", American Programer, 1997, 10(4), pages 13–16

[5] Project failure: Standish Group – Chaos report 1995, http://pmbullets.blogspot.in/2010/04/project-failure-standish-group-chaos.html Accessed on 7 July 2014

[6] Betty H.C. Cheng, Joanne M. Atlee, "Research Directions in Requirement Engineering", Future of Software Engineering (FOSE'07) IEEE, 0-7695-2829-5/07, 2007

[7] J. McDermott and C. Fox, "Using abuse case models for security requirements analysis", In Proceeding of the IEEE Comp. Sec. Appl. Conf., 1999.

[8] S. Uchitel, J. Kramer, and J. Magee, "Negative scenarios for implied scenario elicitation", In Proceeding of ACM SIGSOFT Foundation on Soft. Eng. (FSE), pages 109–118, 2002.

[9] A. van Lamsweerde, "Elaborating security requirements by construction of intentional anti-models", In Proceeding of the IEEE Int. Conf. on Soft. Eng. (ICSE), pages 148–157, 2004.

[10] C. B. Haley, R. C. Laney, J. D. Moffett, and B. Nuseibeh, "The effect of trust assumptions on the elaboration of security requirements", In Proceeding of the IEEE Int. Req. Eng. Conf. (RE), 2004, pages 102–111.

[11] F. Swiderski and W. Snyder, Threat Modelling, Microsoft Press, Redmond, WA, USA, 2004.

[12] C. Heitmeyer, "Applying 'Practical' formal methods to the specification and analysis of security properties", In Proceeding of Information Assurance in Computer Networks (MMMACNS 2001), LNCS 2052, St. Petersburg, Russia, May 2001. Springer-Verlag.

[13] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone, "Modeling security requirements through ownership, permission and delegation", In Proceeding of the IEEE Int. Req. Eng. Conf. (RE), 2005, pages 167–176

[14] R. Crook, D. Ince, and B. Nuseibeh, "On modelling access policies: Relating roles to their organisational context", In Proceeding of the IEEE Int. Req. Eng. Conf. (RE), pages 157–166, 2005.

[15] M. Broy, "The grand challenge in informatics: Engineering software-intensive system", IEEE Computer, 39(10):72–80, 2006.

[16] M. O. Reiser and M. Weber, "Managing highly complex product families with multi-level feature trees", In Proceeding of the IEEE Int. Req. Eng. Conf. (RE), pages 146–155, 2006.

[17] Donald Firesmith, "Common Requirements Problems, Their Negative Consequences, and the Industry Best Practices to Help Solve Them", Journal of Object Technology, Volume 6, No.1, January-February 2007

[18] Paive Parviainen, "Global Software Engineering, Challenges and Solution framework", Thesis of doctor of philosophy, University of Oulu, May 2012

[19] D. E. Damian, D. Zowghi, "An insight into the interplay between culture, conflict and distance in globally distributed requirements negotiations", In Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS '03), Big Island, Hawaii, USA, 6-9 Jan. 2003. IEEE Computer Society

[20] Nikita Nahar, Pujita Wora, Sakthi Kumaresh, "Managing Requirement Elicitation Issues Using Step-Wise Refinement Model", IJASCSE, Volume 2, Issue 5, 2013

[21] http://www.optimusinfo.com/blog/2014/02/04/traditional-vs-agile-software-development.html, Accessed on 8th July 2014

[22] Marian STOICA, Marinela MIRCEA, Bogdan GHILIC-MICU, "Software Development: Agile vs. Traditional", Informatica Economică vol. 17, no. 4,2013

[23] S. Ambler, "Agile Requirements Modeling", 2012 available at http://www.agilemodeling.com/essays/agileRequirements.htm, Accessed on 10 July 2014

[24] Joseph E. Kasser, "Object-Oriented Requirements Engineering and Management", Systems Engineering Test and Evaluation (SETE) Conference, 2003

[25] H. Kaindl, "Is object-oriented requirement engineering of interest?", Journal of Requirement Engineering (Springer-Verlag), Vol. 10(1), pp. 81-84, 2005

[26] Carina Alves, Joao Bosco Pinto Filho, Jaelson Castro, "Analysing the Tradeoffs among Requirements, Architectures and COTS Components", In proceeding of Workshop on Requirement Engineering, Buenos Aires, Argentina, November, 2001

[27] Lu C., Chu W.C., Chang C., Wang C.H., "A Model-based Object-oriented Approach to Requirement Engineering (MORE)", 31st Annual Intl. Computer Software and Applications Conf., (COMPSAC 2007), Vol.1, pp:153-156, 2007.

[28] Anandi Mahajan, Dr. Anurag Dixit, "A Survey of Various Object Oriented Requirement Engineering Methods", COMPUSOFT, An International Journal of Advance Computer Technology, Volume 2, Issue 1, Jan 2013

[29] R. Saranya, "Survey on Security Measures of Software Requirement Engineering", International Journal of Computer Applications (0975-8887) Volume 90 – No 17, March 2014

[30] Matoussi, Abderrahman and Laleau, Regine, "A Survey on Non-Functional Requirements in Software Development Process", Paris: University of Paris, Technical Report TR-LACL-2008-7, 2008.

[31] Philip g. Armour, "Agile… and Offshore", Communications of the ACM, Vol. 50, Issue 1, ACM Press New York, NY, USA, dated: Jan 2007.

[32] N. Ganesh, S. Thangasamy, "Issues identified in the Software Process due to Barriers found during Eliciting Requirement on Agile Software Projects: Insights from India", International Journal of Computer Applications, Volume 16- No.5, February 2011

[33] M. Usman Malik, Nadeem Majeed Chaudhry, Khurram Shahzad Malik, "Evaluation of Efficient Requirement Engineering Techniques in Agile Software Development", International Journal of Computer Applications, Volume 83 – No.3, December 2013

[34] A. Eberlein, F Maurer, "Requirement Engineering and Agile software development", 12th International workshop on enabling technologies, 2003

[35] Brown A. W., Wallnau K.C., "Engineering of Component-Based Systems, Component-Based Software Engineering", Software Engineering Institute, IEEE Computer Society Press, 1996

[36] M. Bichler, K.J. Lin, "Service-Oriented Computing", IEEE Computer, March 2006, pp. 59-68

[37] Erl, T., "SOA Principles of Service Design", Prentice Hall PTR, 2007

[38] M. MacKenzie, et al. "Reference model for service oriented architecture 1.0", Technical report, Oasis, 2006

[39] W3C: Web services description language (WSDL), version 2.0 part 1: Core language. Technical Report, W3C, 2007.

[40] W3C: Simple Object Access Protocol (SOAP), Version 1.2, 2007. http://www.w3.org/TR/soap, Accessed on 6 July 2014

[41] G. Spanoudakis, A. Zisman, and A. Kozlenkov, "A service discovery framework for service centric systems", Proceedings of the IEEE International Conference on Services Computing (SCC'05), Citeseer, 2005, pp. 251–259

[42] K. Zachos, N. Maiden, X. Zhu, and S. Jones, "Discovering web services to specify more complete system requirements," Lecture Notes in Computer Science, vol. 4495, 2007, p. 142.

[43] K. Zachos, N. Maiden, X. Zhu, and S. Jones, "Does Service Discovery Enhance Requirements Specification? A Preliminary Empirical Investigation", Service-Oriented Computing: Consequences for Engineering Requirements, 2006, SOCCER'06, 2006, pp. 2-2.

[44] K. Zachos, N. Maiden, and R. Howells-Morris, "Discovering Web Services to Improve Requirements Specifications: Does It Help?", Requirements Engineering: Foundation for Software Quality, pp. 168-182.

[45] N. Maiden, A. Gizikis, and S. Robertson, "Provoking creativity: Imagine what your requirements could be like", IEEE software, vol. 21, 2004, pp. 68-75.

[46] N. Maiden, S. Robertson, and J. Robertson, "Creative requirements: invention and its role in requirements engineering," Proceedings of the 28th international conference on Software engineering, ACM, 2006, p. 1074.

[47] Ralyte Jolita, "Viewpoints and Issues in Requirements Engineering for Services", In Proceedings of IEEE 36th Computer Software and Applications Workshops COMPSACW 2012, pp. 341-346. IEEE Computer Society

[48] S. Lichtenstein, L. Nguyen, A. Hunter, "Issues in IT Service-oriented requirements engineering," Australasian Journal of Information Systems, vol. 13, 2005, p. 176.

[49] Muneera bano, Naveed Ikram, "Issues and challenges of Requirement Engineering in Service Oriented Software Development", Fifth International Conference on Software Engineering Advances, IEEE 2010 978-0-7695-4144-0, DOI 10.1109/ICSEA.2010.17

[50] C. Rolland, R. S. Kaabi and N. Kraiem, "On ISOA: Intentional Services Oriented Architecture", Proceeding of the 19th International Conference on Advanced Information Systems Engineering, CAiSE 2007, LNCS 4495, Springer 2007, pp 158-172.

[51] J. Cardoso, A. Barros, N. May, and U. Kylau, "Towards a unified service description language for the internet of services: Requirements and first developments," In Proceedings of the IEEE 7th International Conference on Services Computing, 2010, pp. 602-609.

[52] B. Shishkov, J. L. G. Dietz, and M. van Sinderen, "Closing the Business-Application GAP in SOA challenges and solution directions", In 2nd International Conference on Software and Data Technologies, Proceedings, 2007, vol. SE, pp. 333-336.

[53] Teka, Abelneh Y. and Condori-Fernandez, Nelly and Sapkota, Brahmananda, "A Systematic Literature Review on Service Description Methods", In 8th International Working Conference on Requirements Engineering: Foundation for Software Quality, REFSQ 2012, 19-22 March 2012, Essen, Germany.

[54] M. Papazoglou, Web Services: Principles and Technology, 1st ed. Prentice Hall, 2007.

[55] G. Di Modica, V. Regalbuto, O. Tomarchio, and L. Vita, "Enabling re-negotiations of SLA by extending the WS-Agreement specification", In Proceedings of the IEEE International Conference on Services Computing, 2007, pp. 248-251.