A Comparative Study of Software Requirements Tools for Secure Software Development

Mohammad Ubaidullah Bokhari[1] and **Shams Tabrez Siddiqui**[2]

Abstract - Requirement is the foundation of the entire software development life cycle. With proper requirement management a project can deliver the right solution on time and within budget. Requirement elicitation, requirement specification, and requirement validation are important to assure the quality of requirement documentations. All these three activities can be better analyzed by the software requirement tools. There are a number of software requirement tools available both commercially and freely downloadable, which provide a variety and quality of software requirement documentation. In addition, as the vulnerabilities of software increases, system needs an additional requirement for the security aspects which protect the software from vulnerabilities and makes software more reliable. This paper provides a comparative study of requirement tools showing trends in the use of methodology for gathering, analyzing, specifying and validating the software requirements and the result presented in the tables will help the developer to develop an appropriate requirement tool.

Index Terms -Software Security, Software Security Requirement, Requirements Tools, Ontology and Glossary, Software Requirements Specification

1. INTRODUCTION

The history of requirement engineering is not very outdated [2]. In earlier days the requirement phase was not taken seriously which caused many problems for software industry in later phases. Recently the importance of the requirement engineering has been recognized and a lot of research has commenced to generate quality requirements. The process of requirement generation consists of various activities, which includes requirement elicitation which captures the requirement by means of development teams, extending the requirement specification, and finally validates each requirement specification to corresponding user's needs. The requirement engineering is an iterative and co-operative process with an objective to analyze the problem, to document the results in a variety of formats and evaluate precision of the results produced [1]. The specification of requirement should be: complete, correct, consistent, unambiguous, verifiable and traceable.

The non functional requirements, that are separate from procedural requirements includes : maintainability, portability, reusability, reliability, security and others does neither have any exact specification nor are there any metrics for specifying objectives of requirements [2]. In addition the functionality and its associated functional requirement widely varies between applications, especially between different application domains [3]. However, the same cannot be illustrated in the case of non functional requirements. The security criteria (authentication, privacy, authorization, integrity etc.) of some applications exhibit to less variation.

The system in our dictionary is concerned with an entity that interacts with the environment (hardware, software, human, and physical world with its natural phenomena). Its fundamental characteristics are computation and communication which may be characterized by fundamental properties like: performance, behavior, dependability, and security. Performance means up to what extent the system satisfies the user needs. Behavior includes what the system does to implement its function and is described by a sequence of states. Dependability means to avoid service failures that are more frequent and more severe than is acceptable. It is an integrating concept that encompasses the following attributes [4]:

1. Availability: readiness for correct service.
2. Reliability: continuity of correct service.
3. Safety: absence of catastrophic consequences on the user(s) and the environment.
4. Integrity: absence of improper system alterations.
5. Maintainability: ability to undergo modifications and repairs.

Security is very similar to dependability except additional attribute confidentiality (absence of unauthorized disclosure of information) which has an edge over the dependability attributes. Security comprises attributes of confidentiality, integrity and availability (CIA).

In the software development processes both designers, and requirement engineers play an important role to develop quality software. Many techniques have been proposed. There are specific requirement elicitation techniques, such as interviewing, questionnaire, or storyboarding techniques, for the specification of the requirements, such as scenarios or use case modeling, and for the validation of the elicited requirements, such as prototyping.

The objective of this work is to show the advantages and disadvantages of the current requirements tools, present a palette of requirements engineering techniques which could aid requirement engineer in their work. In addition, the comparison presented should help in the continuous process of improvement of the existing software requirement tools in order to focus more on requirements engineering, and therefore

[1]Chairman, Department of Computer Science, Aligarh Muslim University, Aligarh
[2]Research Scholar, Department of Computer Science, Aligarh Muslim University, Aligarh.
E-mail: [1]mubokhari2004@yahoo.co.in, mubokhari@rediffmail.com, [2]ksamtab@rediffmail.com and [2]kk_siddiqui@yahoomail.com

contribute to improve the quality of the requirement capturing that are built with these methodologies. The present work gives a survey and a comparative study of the current approaches available in the requirement tools that use different methodologies and model to handle requirements engineering. For that reason, we outline the requirements engineering process and an overview of classic requirements engineering techniques in § 2. The brief description includes the most commonly used techniques to capture, define and validate the requirements of a system. In § 3, the main requirement engineering methodologies are described including requirements specification, that in different degree of detail include requirements specification. This section also includes a classification of requirements. In § 4, these approaches are compared from different points of view. Finally § 5 presents some conclusions and suggestions for future.

2. SOFTWARE REQUIREMENT ENGINEERING

Software requirement is defined as a condition or capability that must be met or fulfilled by a system to satisfy a contract, standard, specification, or other formally imposed documents (IEEE standard 610.12-1990). The requirement engineer shall address the following things (IEEE standard):

1. Functionality: What the software supposed to do?
2. Performance: How does the software perform the user needs?
3. Attributes: What are the portability, maintainability, correctness, and security considerations.
4. External Interface: How does the system interact with the environment?
5. Design constraints: Are there any required standards in effect, policies of database, resource limit, operating environments etc.?

Although, there are a number of techniques available to perform the task of requirement engineering, but a standardized process supporting the requirement engineering and guaranteeing the quality of the software requirement is still lacking. The process of requirement engineering consists of mainly three activities which are described as follows:

2.1 Requirement Elicitation

The process of requirement gathering is by asking stakeholders to describe their views. This is a complex procedure because it requires several groups of participants with different backgrounds. The most widely used techniques for requirement elicitation are:

1. **Interview.** Frequently used to understand the problem domain and objectives of the application to be developed. Interview is not easy to perform. It requires an experienced interviewer who needs to have the ability to choose the most appropriate interviewees [5].
2. **Joint Application Development (JAD).** An alternate method of interview in which a group of participants of all stakeholders project i.e. analysts, designers, users, system administrator, and customers are involved [6]. It has several advantages over the interview technique as it saves time.
3. **Brainstorming.** A type of group meeting similar to JAD which consists of collection of non evaluated ideas and information of stakeholders of the projects [7]. Brainstorming is easier as compared to JAD because it requires less work in the group and provides a better overview of the software requirement.
4. **Questionnaire and Checklist.** A technique in which a well prepared question has a limited choice of options and a concrete answer is possible. The main drawback of this technique is that an analyst should have certain knowledge about the problem domain and the application to be built in order to prepare the questionnaire and checklist.
5. **Case Modeling.** A technique was developed to define requirements more than for capturing them [8]. It is used to define actors, use cases and relationship between them.
6. **Scalability** is a method used by designer to provide an option to increase the requirement of an end user of a system if the business expands.

2.2 Requirement Specification

To improve accuracy and relevancy of the software requirement, the most widely used techniques are:

1. **Use Case Modeling** is a technique to define requirement although it is also used in elicitation of requirement. The main disadvantage of this method is that it is an ambiguous technique when defining complex requirements [9, 10].
2. **(TRS) Tradition Requirement Specification** is an ambiguous technique to define requirement in natural language without any kind of rule.
3. **Templates** are used to describe the requirements in natural language but in structured format. Templates are a table whose fields have a predefined structure and are filled by the development team using the user's terminology.
4. **Glossary and Ontology** are used to define the terminology related to the terminology used in the software project. Ontology is the method used to define the relationship of the concepts used in the project.
5. **Prototype** is a method to provide a context within which users are better able to understand the system they want to be built.

2.3 Requirement Validation

To improve accuracy and completeness of the software requirement. The following are the techniques used for requirement validation process:

1. **Audit** checks the results presented in the review documentation and compares it to the available checklist. It provides only a partial review of the information and results.
2. **Walkthrough** consists of reading and correcting the documentation and validates only the good interpretation of the information.
3. **Prototyping** is a technique used to validate the requirement using an existing software requirement tool.

3. SECURITY REQUIREMENT ENGINEERING

Security policy means to protect the software system by capturing secure software requirement of the system. Jan Jurjens [11] suggested some security requirements which are discussed below:

3.1 Fair Exchange

Fair Exchange requirement postulates that the trade performed by e-commerce is fairly treated and prevented by cheating from either side. (e. g. the buyer or supplier should be able to prove the payment is made or goods supplied and to reclaim the money if the payment or goods are not delivered.

3.2 Non-repudiation

Non-repudiation security requirement supports the fair exchange, which means that action cannot be subsequently denied.

3.3 Role-based Access Control (RBAC)

Role-based access control security requirement play an important mechanism for controlling access to protect assets. It keeps permission manageable, with a large or frequently changing user-based software system

.

3.4 Secrecy and Integrity

The two most important security aspects are Secrecy (Confidentiality) and Integrity. Secrecy means the resources can be used only by legitimate party. Integrity of data means that it should be modified only by an authorized person.

3.5 Authenticity

The third main security requirement is authenticity. Authenticity can have two types, *Message authenticity* and *entity authenticity*. Message authenticity means that one can trace the data back to what its original source was, at some point in the past. Entity authenticity means it ensures the party who can identify participants in a protocol, and in particular make sure that the party has actually actively participated in the protocol at the time.

3.6 Freshness

A message can be treated fresh if it has been created during the current execution round of the system under consideration and therefore, cannot be a replay of an older message by the adversary.

3.7 Secure Information Flow

Security level can have different rules. One usually considers two security levels: high and low. High means highly sensitive or highly trusted whereas low means less sensitive or less trusted. Where trusted parts of a system interact with untrusted parts, one has to ensure that there is no exchange of data from trusted parts to untrusted parts. To ensure this no down flow policy, low data may influence high data, but not vise versa. The opposite of the condition no up-flow, enforces that parts of a system not trusted may not directly influence high data. High data may influence low data, but not for the opposite case.

3.8 Guarded Access

One of the principal security requirements is access control, which means that only a trusted user can have an access to a security based system.

4. SOFTWARE REQUIREMENTS TOOLS

The requirement specification obtained after requirement analysis used throughout the software development lifecycle, must be verified because the quality of the requirements is of utmost importance to deliver the right product. A number of software requirement tools which verify the qualities of the software requirements are given below:.

4.1. RequisitePro

The IBM Rational RequisitePro solution is a widely used and familiar Microsoft word tool to ease requirements based on use case model for software development project teams who want to improve the goals, enhance collaborative development, reduce project risk and increase the quality of applications before deployment [12].

Features:

1. The requirement in word documents are dynamically linked to supplementary requirements information stored in a database. They contain live requirements and allow remaining in a familiar Microsoft word environment to modify requirements.
2. From views into the database, it can be prioritizing link requirements and track changes and show requirements that can be affected by upstream and downstream change.
3. Enables detailed attribute customization and filtering to maximize informative value of each requirement
4. Connect requirements to use case model, enabling instantaneous access to use case specification from use-case diagram as well as visibility into requirements information.
5. Performs project-to-project comparisons using exportable XML-based project baselines.
6. Integrates with multiple tools and teams in the IBM Software Development Platform to improve accessibility and communication of requirements.

Disadvantages:

1. Since it connects requirements to use case model only and does not consider a misuse case. There may be some hole of intrusion present and the system may be considered as insecure.
2. Since a checklist can provide a better and tested requirements and RequisitePro does not have a checklist to ensure the criteria of the requirements.

3. Scalability is also an important non-functional requirement of software. RequisitePro has no provision for scalability according to project size.
4. Ontology and Glossary are the best practices to support the software development team throughout the software development life cycle. RequisitePro does not have any online glossary/ontology to define industry terms project references, corporative languages etc.
5. The key to understanding the problem are the process and project management requirements that reliability and maintainability which provide the ability to consider the different perspectives of the various contributors to the system development effort and to capture and relate the entire requirement. These facilities are not available.

4.2 CaseComplete

CaseComplete is a tool developed by Serlio Software to manage, share use cases and requirements based on Microsoft Word, CaseComplete helps to write use cases and requirements faster and easier that have excellent compliance with use case standards for a novice or an expert user whether working on solo system or a part of diverse team [13].

Features:

1. The requirement reports generated in word documents are dynamically linked with other requirements documented and stored in other place.
2. The requirement report is integrated with other phases of software development life cycle.
3. The report is generated in Microsoft Word and HTML formats of individual and complete requirement of the system
4. Generates the test plans, project plans and UML models directly from use case model.
5. Provides an index of glossary items which help to understand the terminology used in the system.
6. Covers the non-functional requirement specification.

Disadvantages:

1. There might be a need to increase the project team size. CaseComplete does not include scalability.
2. A checklist nowadays provides a better, tested requirement and CaseComplete does not contain any checklist. Therefore one cannot say that the gathered requirements have to be testified and have quality.
3. Since it has an export/import feature and can be hyperlinked with other application, an unauthorized user might be send or capture data. This tool is considered as a weak tool by means of security.
4. Since a wizard is considered as an easy method to just guide the requirement engineer to do their specific goal, CaseComplete can not be customized using any GUI or by using any wizard.

4.3 Analyst Pro

Analyst Pro is a tool for requirements management, tracing and analysis developed by Goda Software Inc. With Analyst Pro, requirements can be traced with any lifecycle software model e.g. waterfall, RUP, spiral. It also provides integrated configuration management to simplify the development process. It can be easily installed and deployed to geographically dispersed teams to collaborate on specification, analysis and project management [14].

Features:

1. Requirements Specification and Tracking – AnalystPro quickly establishes multidimensional traceability links with all project artifacts.
2. Scalable from 1 to 250 users.
3. Repository (for Non-Functional Requirements Objects) – Analyst Pro provides a repository for non-functional requirements objects. UML and other models created by external tools can be saved to the repository for sharing, collaboration, and configuration management, and for linking them to requirements and specifications.
4. Configuration Management – Analyst Pro simplifies the development process by providing integrated configuration management for project artifacts. Analyst Pro allows to baseline and lock project artifacts.
5. Other features include: Reusability of project settings and specification templates using project templates, control of access by creating user groups with different privileges; ability to assign a requirement or other task to team members and review their progress; built-in diagramming editors for creating project diagrams; easy generation of system documentation and change history reports, baseline comparison, traceability reports and status reports etc.

Disadvantages:

1. As it has an export/import facility, it can be interfaced with other applications. There might be a chance that the data may be accessed by other user.
2. As it is concerned with use case model requirements only and does not consider a misuse case, there are some holes of intrusion and system may be considered insecure.
3. The glossaries are the best practices and support the software development team throughout the software development life cycle. AnalystPro does not have any online glossary to define any industry term, project references, corporative languages etc.
4. It does not have any checklist to verify the criteria of requirements and a checklist is always facilitates to deliver quality and tested requirements.
5. Models and simulation are key components to understand all the relevant issues, early identification of risk areas and finding out alternate solutions produce quality system at lower cost. These facilities are not available with this tool.

4.4 Optimal Trace

The Optimal Trace developed by SteelTrace products provides a pragmatic approach for organizations to quickly and easily capture business and system requirements with a 40% per project ROI with little training or deployment expenditure necessary. Unlike traditional RM tools, Optimal Trace takes a more structured view of requirements breaking them into Functional (in the form of a use case like storyboard structure of main flow, alternative flows etc.) and Non-functional requirements (qualities and constraints). Optimal Trace automatically generates graphical flows directly form text and maintains text and graphics in lockstep. Optimal Trace Professional is a single user version while Optimal Trace Enterprise is a multi-user variant [15].

Features:

1. Optimal Trace offers native integrations from Optimal Trace into UML modeling tools Rational Rose and Borland Together Solo and Control Center. The integrations are bi-directional with UML Use Case and Activity diagrams are automatically generated.
2. Optimal Trace Enterprise is designed for teams who are collaborating on requirement gathering and capture by updating and sharing a project.
3. It is tightly integrated with tools from the following leading vendors in the UML/MDA modeling space; Compuware (Optima1J Integration), IBM (Rational Rose Integration) and Borland (Together Integration).
4. Communication is easy with Optimal Trace's automated document generation and a selection of pre-canned templates that are fully customizable to company–specific standards and processes to ensure a high-quality requirement.

Disadvantages:

1. It does not contain any checklists to verify the criteria of requirements. Therefore, the requirements captured by the tools are not considered to be quality requirements as the requirements are not tested.
2. Since the requirement is concerned with use case model only a misuse case model is completely ignored. The requirements are not secured.
3. As it has an interface facility with other applications, there might be a chance that the data may be accessed by other unauthorized users. Therefore, it may be considered as a weak tool with regard to security aspects.
4. There might be a chance to increase the project team size and Optimal Trace has no feature of scalability according to team size which may create problem.
5. The glossaries are the best practices and support the software development team throughout the software development life cycle and it does not have any online glossary to define any industry term, project references, corporative languages etc.

4.5 DOORS

DOORS (Dynamic Object Oriented Requirements System) is an Information Management and Traceability (IMT) tool developed by Telelogic Inc.. Requirements are handled within DOORS as discrete objects. Each requirement can be tagged with an unlimited number of attributes allowing easy selection of subsets of requirements for specialist tasks. DOORS includes an on-line change proposal and review system that lets users submit proposed changes to requirements, including a justification. DOORS offer unlimited links between all objects in a project for full multi-level traceability. Verification matrices can be produced directly or output in any of the supported formats including RTF for MS-Word, Interleaf and Frame Maker. The DOORS Extension Language (DXL) is a high level language that provides access to virtually all DOORS functions for user extensions and customization [16].

Features:

1. A comprehensive support for recording, structuring, managing and analyzing requirement information.
2. User friendly Interface.
3. Import and export facility with other documents.
4. Complete two ways traceability across the development life cycle.
5. Unparallel integration with third party tools like Mercury and Matrixone.
6. Improved security control through the use of passwords, and timeouts which "lock up" DOORS after a specified period of inactivity.
7. Scalability for any size project with any number of users in any location.
8. New templates to make document generation easier have been added to the DOORS template library. New templates include ISO 12207, ISO 6592 and IEEE software standards.

Disadvantages:

1. Ontology and glossary are the best practices and support the software development team throughout the software development life cycle but it does not have any online glossary.
2. Since it has an export/import facility and can be interfaced with other applications, there might be a chance that the confidential data may be accessed by other users.
3. A checklist provides a better quality and tested requirements. However, DOORS does not contain any checklist.

4.6 GMARC (Generic Model Approach to Requirements Capture)

This tool developed by Computer System Architects Ltd. incorporates a fully developed Requirements Engineering Methodology and provides rapid elicitation of requirements using a generic approach to enhance re-usability and encourage standardization across projects [17].

Features:

1. The requirements can be directly elicited from the minds of the expert.
2. Traceability of requirements hierarchically, historically and inter-task as well as inter-document.
3. Identification and correction of subjective requirements.
4. The goals and constraints are separated.
5. Generic approach enhances re-usability and encourages standardization across projects.
6. Ease of modifiability of requirements documents with automatic adjustment of knock-on consequences.
7. Automatically generate data flow diagram models of functional aspects.

8. Ability to verify dynamic viability of system being specified via animation.
9. Automatic interchange of requirements information between models and specifications.
10. Ability to confine text output to any viewpoint for any application aspect for any layer of support at any level of detail or any combination.
11. Generate a powerful documentation structuring and filtering facilities.
12. Standard generic text interface simplifies linking to any other package.
13. User friendly Human Computer Interface.

Disadvantages:

1. The size of the project may increase and GMARC does not have any feature of scalability according to team size which may create a problem.
2. A checklist is nowadays provides a better quality and tested requirements and GMARC does not contain any checklist therefore one can not say that the gathered requirements have quantity and quality.
3. As the requirement is not concerned with use case model and a misuse case model. Therefore, we may say that the requirements are not perfect regards with secured aspects.
4. The glossaries are the best practices and support the software development team throughout the software development life cycle and it does not have any online glossary to define any industry term, project references, corporative languages etc.
5. There might be a chance to increase the project team size and GMARC does not have any feature of scalability according to team size which may create a problem.

4.7 Objective

Objectiver has been developed by Cediti and designed by RE practitioners to enable real requirements engineering. The tool relies on Kaos, a goal driven methodology and enables users to have a global overview on the system and a systematic link between all the models representing the system. Analysts have the possibility to draw diagrams and to define concepts (like goals, requirements, agent, entities, events, relationships, actions,) and relationships over those concepts (like refinement, conflict, operationalisation, responsibility, capability, performance, specialization, causes and so on). Diagrams can be explained with text documents including references to concepts elicited in the diagrams. All these pieces of information can then be put together to generate a requirements document compliant with predefined standards [18].

Features:

1. It builds a requirement model to describe the problem by defining and manipulating the relevant concept including queries.
2. It justifies the requirements by linking them to higher-level goals.
3. Provides a consistent and complete glossary of all the problems related terms.
4. It produces well structured, self-contained, motivated, easily understandable, standard requirements automatics generated documents.
5. It provides highly effective way to communicate about the requirements through multiple views on documents with easy navigation.
6. It ensures traceability from requirements to goals and checks the completeness and consistency of the requirements.

Disadvantages:

1. The size of the project may increase according to need of the stakeholders and Objectiver has no scalability feature which may create problem.
2. It does not have any export/import features to other software tools. Therefore, it may create problem if someone has to capture the requirement written in other tools.
3. It is not base on either use case or misuse case, therefore, from the security point of views it is hard to identify the vulnerability holes and attackers ideas.

4.8 RDT (Requirements Design & Traceability)

RDT is a software requirement, design, and traceability tool is developed by Igatech System Pvt. Ltd. It supports several mechanisms to aid the user in requirements analysis and identification. It can be used to formulate and generate specification before the specification is issued as a contract or request for tender. These include a parser that imports text documents then identifies requirements by key words and structure. The tool provides functionality for deriving, allocating and assigning requirements and acceptance test procedures. Requirements can be traced from top level requirements down to the lowest level requirements. The tool is able to classify/categorize requirements during identification using requirements attributes. It can also be used to show the design and traceability of requirements throughout the development of cycle of the contract. RDT is able to generate documentation directly into MS Word, including requirements and test specifications, requirement allocation matrices, parent-child relationships and design documents [19].

Features:

1. Revision tracking and baseline allocation including proposals for changes.
2. Workgroup access privileges to control user access down to individual records.
3. Comprehensive online context sensitive help.
4. Network accessible for multi user database access-up to 255 concurrent users.
5. Change Proposal Management, which enables a change proposal to be identified, and any data which will be added, changed, or deleted as a result of it being accepted.
6. Revised Import/Export, allowing sections of the database to be exported, including relationships with other requirements, tests and derivations.

7. Check-In Check-Out, enabling the sharing of data between different sites, and the ability to collate this data back to the master database.
8. User Defined Attributes gives users the ability to name their own unique attributes.
9. Document View Editing, providing a word processor style view of document data.
10. Automated requirements capture and syntax parsing directly from existing documents.
11. Improved User Interface. All data viewing windows are now available with multiple instances, enabling concurrent views of different data

Disadvantages:

1. A checklist always provides a quality, better and tested requirements. RDT does not contain any checklist therefore the gathered requirements have not to be quantified and have a better quality.
2. As the requirement is not concerned with use case model and a misuse case model. Therefore, we may say that the requirements are not perfect as concerned with secured aspects.
3. Since it has an export/import facility and be interfaced with other application. There might be a chance that the confidentiality of the data may be access by other user

4.9 RDD-100

RDD-100 is a Requirements Driven Development (RDD) software suite developed by Holagent Corporation. The RDD-100 uses several mechanisms to aid the user in analyzing and identifying requirements. These include a parser tool that can be defined and developed to help the user identify single or compound requirements. RDD captures and trace the requirement using its Element Relationship Attributes (ERA) and categorize them in a specific manner, where each source document, and the text for each requirement, is stored as a separate element. Graphical hierarchies show how individual pieces of data relate to each other and trace back to their sources [20].

Features:

1. RDD-100 provides the user the capability to interactively manipulate and data through a variety of diagrams including Behavior Diagrams, Hierarchical Views, Functional Flow Diagrams, N2 charts, IDEF0 and Data Flow Diagrams.
2. The report writers available in RDD-100 provide the users to manually create and identify requirements through several types of views.
3. Access Control to the data stored within the system design database can be managed within RDD-100 and can be determined by the rules of the user's process.
4. Data stored in RDD-100 can be shared with tools such as UML, hardware tools, scheduling tools, word processing tools, project management tools etc.
5. RDD-100 has online documentation that includes a user's guide and on-line help.
6. **Disadvantages:**
7. The size of the project may be increase and RDD-100, has not any feature of scalability according to team size which may create problem.
8. A checklist ensures the quality of requirements. RDD_100 does not have any provision of checklist therefore one can not say that the gathered requirements have to be tested.
9. As the requirement is not concerned with use case model and a misuse case model. Therefore, we may say that the requirements are not perfect as concerned with secured aspects.
10. As it has an interface facility with other application. There might be a chance that the data may be access by other unauthorized user. Therefore, it may be considered as a weak tool as concerned with the security aspects.

4.10 Requirements Traceability Management (RTM)

RTM developed by Serena Software Inc. supports multiple users working on the same requirements at the same time by implementing locking control on a requirement-by-requirement basis. Serena RTM is the only tool that supports the critical capabilities, at the object level on UNIX and PC platform using a standard database. RTM's toolset supports the ability to capture graphical information as traceable requirements objects. A class definition tool is included that allows the user to model any type of hierarchical project data (requirement document, hierarchies, system element structure and WBS). Once the hierarchy is defined generic relationships can also be established to allow cross-reference link information to be established between any active data item. Serena RTM is the only Oracle-based tool designed to manage all of the data for the development. It is also called as Engineering Information Management (EIM) tool, or Development Data Management (DDM) Tool, RTM allows you to organize and manage Critical Development-related data [21].

Features:

1. RTM has ability to intuitively organize and manage the information such as requirements, design, test, schedules changes, defects etc.
2. RTM provides online collaboration by all stakeholders, regardless of location.
3. Ability to remotely edit an MSWord document, which allows user to work on the documents offline.
4. Visibility into the state of each phase of development
5. RTM manages change at all levels of development through email notification of changes to other users.
6. RTM is a user interface both word and web, that are industry standards and commonly used in everywhere.
7. RTM is build on Oracle and therefore it offers Oracle advantages such as role base access control, guarded access, authenticity etc.

Disadvantages:

1. The size of the project may be increase and RTM, has not any feature of scalability according to team size, which may create problem.
2. A checklist always provides a quality, better and tested requirements. RTM does not contain any checklist

therefore the gathered requirements have not to be quantified and have a better quality.
3. Since it has an interface facility with other application. There might be a chance that the confidentiality of the data may be access by other user.
4. The Glossary are the best practices and support the software development team throughout the software development life cycle and it does not have any online glossary to define any industry term, project references, corporative languages etc.
5. It does not base on either use case or misuse case, therefore, from the security point of views it is hard to identify the vulnerability holes and attackers ideas

4.11 Reqtify

Reqtify is a requirement-monitoring tool developed by TNI-Software. It is the most effective solution for project teams to capture requirements from any source and easy to use for traceability and impact analysis, enabling quality development in both hardware and software projects [22].

Features:
1. Detection of requirement changes.
2. Graphical view: zoom, moves and resizing of documents.
3. Capture of requirements attributes, references, links, etc. at any level (High-level, Low-level)
4. Reports Generation & Customization capabilities
5. Interface with other tools like Processing tools, requirement tools, UML tools, verification tools etc.
6. Filter creation for more accurate analysis.

Disadvantages:
1. Glossary and Ontology are the best practices and support the software development team throughout the software development life cycle and it does not have any online glossary to define any industry term, project references, corporative languages etc.
2. It has the provision of interface with other software; therefore, the data of the software requirement can be accessed by some malicious user. Thus there is a chance of information disclosure.
3. The size of the project may be increase and Reqtify has not any feature of scalability according to team size which may create problem
4. Since a checklist can provide a better and tested requirements and Rectify does not have a checklist to ensure the criteria of the requirements.

4.12 IRqA

IRqA is not a specific requirement-engineering tool, but also it focused on information exploitation, which provides support to the entire requirement engineering cycle [28].

Features:
1. Requirement can be captured by manual and automatic from MS Word documents.
2. Graphical representation of concept models: class diagram (UML) and ER diagrams.
3. Classification criteria defined by the user, specification is being consistently checked.
4. Validation of requirements implementation in service.
5. Multiple specifications in domain and/or blocks.
6. Full end-to-end traceability from user requirements to the detailed design, implementation and test.
7. Integration with Object Oriented design tools with XML/UML export/import facility.
8. Reports defined by user in order to create documents based on industrial standards or existing templates in each organization.

Disadvantages:
1. The size of the project may be increase and IRqA has not any feature of scalability according to team size.
2. Glossary help and support the software development team throughout the software development life cycle and it does not have any online glossary to define any industry term, project references, corporative languages etc.
3. It has the provision of interface with other software; therefore, the data of the software requirement can be accessed by some malicious user. Thus there is a chance of information disclosure.

4.13 TcSE (Teamcenter Systems Engineering)

TcSE is a requirements management tool developed by UGS Inc. includes both Requirements Management and System Architect licenses. The Systems Architect solution gives the people responsible for planning the integrated mechanical, electrical and software product design a powerful tool to create and communicate requirements. The Requirements Management solution delivers product requirements to all of the entitled users who participate in your product lifecycle. Teamcenter brings your customers directly into your extended enterprise and reflects their concerns from the start of your product lifecycle to its conclusion [23].

Features:
1. An intuitive user interface that looks and acts a lot like Windows Explorer and Outlook.
2. Microsoft Office integrations allow users to interact with requirements information directly from their desktop
3. Document importing and exporting for requirements capture and generation.
4. Multi-user group environment that enables users to view and work on requirement concurrently in a controlled way.
5. Linking and tracing mechanisms like a summary requirement to a specific paragraph in the source document from which is was extracted
6. Security protections allow administrators to control user access, information access, and modification privileges.

Disadvantages:
1. Ontology and Glossary are the best practices and support the software development team throughout the software development life cycle and it does not have any online glossary to define any development terminology, project references, corporative languages etc.

2. The size of the project may be increase and TcSEe, has not any feature of scalability according to team size, which may create problem.
3. It has the provision of interface with other software; therefore, the data of the software requirement can be accessed by some malicious user. Thus there is a chance of information disclosure.
4. A user always wants the tools which are easy to use. A graphical representation always considered a better option for any requirement tools. TcSE does not has such feature

4.14 Code Assure

Code Assure Solo is the industry's first enterprise-class application security tool designed specifically to meet the needs and cost requirements of individual developers, project managers and security architects. It is the only tool that provides a comprehensive, process-oriented solution for identifying, assessing and remediation of software vulnerabilities throughout the development lifecycle [25].

Though CodeAssure Solo is purpose-built to meet the needs of individuals, it offers the same comprehensive security analysis capabilities that support entire development and security teams. With CodeAssure Solo, anyone – regardless of company size or budget – can benefit from the industry's most accurate analysis tool.

Features:

1. CodeAssure identifies and remediate the software vulnerabilities early in SDLC.
2. Deploy in an Eclipse environment and on existing projects
3. Achieve high acceptance rate for code without security flaws
4. Integrates into existing processes and systems.
5. Reduces the time and costs associated with analysis, remediation, and deployment of secure application.
6. Stakeholders are continuously informed of the current status of application security and policy violation.

Disadvantages:

1. A checklist ensures the quality of requirements. CodeAssure does not have any provision of checklist therefore one can not say that the gathered requirements have to be tested.
2. Ontology and Glossary are the best practices and support the software development team throughout the software development life cycle and it does not have any online glossary to define any development terminology, project references, corporative languages etc.
3. The size of the project may be increase and CodeAssure, has not any feature of scalability according to team size which may create problem.
4. A graphical user interface is very handy to use for a novice requirements engineer. CodeAssure has not such feature.

5. COMPARATIVE STUDY

The comparative study of the above requirements tools are based on two factors. The first one is analysis of the requirements simply gathered for the functional requirement, and second one is analysis of the requirements gathered for the security point of view as shown in the Table-1 and Table-2. The first one is said to be product oriented where as the second one may be regarded as security oriented.

All above requirement tools are compared with the parameters discussed in Software Requirement Engineering and Security Requirement Engineering sections earlier. In both section there are eight parameters and each parameter have been assigned (100/8) 12.5 points. The right ticks (√) show that the tools fulfill the parameters in which it exists. The total number of (√) are counted and multiplied by 12.5 and its total value is presented in Table-3. The tool having maximum points is considered as best one.

5.1 Product Oriented

Product oriented is the approach to describe the steps to be followed in order to perform the capture the requirements specification, elicitation and validation.

4.3 Security Oriented

Security oriented describes the techniques to be applied during the process of requirement gathering in order to satisfy the user non functional needs with respect to security aspects

6. POSSIBLE SHORTCOMINGS

There are possible shortcomings to our study. First, we want to stress that we do not have any right to compare the qualities of software products. Therefore, we cannot know if certain areas have been left out because of deliberate decision or lack of information or knowledge. As a consequence we do not judge the requirements tool as good or bad, but rather analyze the functionality of these tools. Second, the research work is based on the software freely downloaded or the documents available in the websites. The trial version of the software has some restricted privileges; therefore, it is hard to judge the quality of that tool.

7.0 CONCLUSION

In this paper we have presented the state of the art of requirements engineering both functional and non security and discuss their usability according to process and product. According to Table-3, we can conclude that DOORS are the one of the best requ
irement tool satisfying the both functional and non-functional requirements, whereas for the security point of view CodeAssure is the best among all the tools. We also advise to add checklist and glossary features and make CodeAssure the best security requirements tool.

As a result of our study, we still advise that there are a great potential in the field of the requirement engineering to gather, elicit and validate the requirements with respect to the functional and security aspects. We hope that the result presented in the tables will help developer to develop an appropriate requirement tools to overcome all the drawbacks as we mentioned in this particular work and which will be beneficial for the functional and security point of views.

REFERENCES

[1]. Lowe D. Hali, W Hypermedia and the Web application Engineering Approach, John Wiley & Sons. 1999.

[2]. Adam Sachitano, Richard O. Chapman, “Security in Software Architecture: Case Study”, Proceedings of 2004 IEEE Workshop on Information Assurance, United States Military Academy, West Point, NY 10-11 June, 2004.

[3]. Donald Firesmith: “Specifying Reusable Security Requirements”, *Journal of Object Technology*, vol. 3, no. 1, January-February 2004.

[4]. Algirdas Avi_zienis, Fellow, IEEE, Jean-Claude Laprie, Brian Randell, and Carl Landwehr, “Basic Concept and Taxonomy of Dependable and Secure Computing”, IEEE Transactions on Dependable and Secure Computing, Vol. 1, No. 1, January-March 2004.

[5]. Pan, D., Zhu, D., Johnson, K. Requirements Engineering Techniques. Internal Report. Department of Computer Science. University of Calgary. Canada., 2001

[6]. Livesey D., Guinane T, “Developing Object-Oriented Software, An Experience-Based Approach (IBM's OOTC)”, Prentice Hall, 1997.

[7]. Raghavan, S., Zelesnik, Ford, G. Lectures Notes of Requirements Elicitation. Educational Materials CMU/SEI-94-EM-10., 1994.

[8]. Jacobson, I. “Modeling with Use Cases: Formalizing Use Case Modelling”. Journal of Object-Oriented Programming, 1995.

[9]. Insfrán, E., Pastor, O., Wieringa, R “Requirements Engineering-Based Conceptual Modeling”. Requirements Engineering Journal, Vol 7 (1)., 2002.

[10]. Vilain, P., Schwabe, D., Sieckenius, C. “A diagrammatic Tool for Representing User Interaction in UML”. Lecture Notes in Computer Science. Proc. UML’2000. York, England., 2000.

[11]. Jan Jurjens, “Secure Systems Development with UML”, Springer-Verlog, 2005.

[12]. www-306.ibm.com/software/awdtools/reqpro.

[13]. http://www.serlio.com/casecomplete

[14]. http://www.analysttool.com

[15]. www.compuware.com/products/optimaltrace

[16]. http://www.telelogic.com/corp/products/doors

[17]. http://myweb.tiscali.co.uk/gmarc/general%20features.htm

[18]. http://www.objectiver.com

[19]. http://www.igatech.com/rdt

[20]. http://www.holagent.com

[21]. http://www.serena.com/Products/rtm/home.asp

[22]. http://users.reqtify.tni-software.com/?p=home

[23]. www.ugs.com/products/teamcentre

[24]. http://sesa.dit.unitn.it/sttool/

[25]. http://www.securesoftware.com/products

Tools	Product	Security
RequisitePro	62.5	25.0
Case Complete	72.5	25.0
Analyst Pro	50.0	37.5
Optimal Trace	62.5	12.5
DOORS	72.5	62.5
GMARC	50.0	12.5
Objectiver	62.5	12.5
RDT	62.5	12.5
RDD-100	50.0	25.0
RTM	37.5	50.0
Reqtify	50.0	25.0
TcSE	37.5	62.5
Code Assure	25.0	87.5
IRqA	50.0	12.5

Table 3: Details of Product and Security Technique