CHAPTER-4

REQUIREMENT ENGINEERING

Requirements engineering is the discipline concerned with understanding the externally imposed conditions on a proposed computer system, determining what capabilities will meet these imposed conditions and documenting those capabilities as the software requirements for the computer system. Requirements were the desired characteristics for the software to be being developed. The first activity in most of the projects was the identification and documentation of the requirements. The development of large, complex systems presents many challenges to systems engineers. Foremost among these was the ability to ensure that the final system satisfies the needs of users and provide for easy maintenance and enhancement of these systems during their deployed lifetime. These systems often change and evolve throughout their SDLC. This makes it difficult to track the implemented system against the original and evolving user requirements. Various studies have shown that roughly half of the application errors and vulnerabilities can be traced to requirement errors and deficiencies. Thorough proper documentation and securely managing requirements can help in better developing quality applications.

In present work the aim was to measure the impact of software design on software security, so the security model for an object oriented design (SMOOD) was proposed. In this chapter the requirement related to the SMOOD model development was collected. Based on the collected requirement, software requirement specification document was prepared which could guide for the model development.

4.1 Requirements Gathering:

The proposed system is a research work, so most of the requirements were gathered from the literature available. A requirement gathering was carried out by referring research papers, article, website and books. The major reason of failure of the secure software is either misunderstanding of the security aspect in software development life cycle or missing the requirements in the development phase which could stated in the requirement phase. The available research works related to same problem were considered and limitations of present research work were listed in literature review. In our proposed model, an attempt was made to overcome all the limitation present in
current research finding by means of a practical solution which could estimate security at design stage. Hence, in this work most of requirements were collected from limitations of the current research findings.

4.2 Requirements Analysis:

Requirement analysis is the process of analyzing needs of the clients in order to arrive at a definition of the requirements. If elicitation activities were intended to gather constraints then analysis aims to extend our understanding of the constraints and client needs. For measuring the impact of software design on software security, the software design needs to be checked. While checking the design, flaws and weaknesses need to highlight.

Software design is a collection of different design properties. While in design phases the designer has choice to select required design properties. If the designers make the proper selection it will lead to the strong design and vice versa. Designers should have knowledge the impact of design properties on security. For that the designer should be a security expert which normally or practically not possible. But if the designer has any tool or model which could work as security expert then this definitely guides the designer in right direction.

In this research work, an attempt was made to check the class diagram from security point of view. Class diagram gives the structural view of the system and all the necessary information like classes their available relationship. The proposed model should check the security of design and gives the result in the specific measure instead of providing guideline in design phase. The focus was given on the design stage and tried to find weakness or flaws of design phase. In addition to that the focus should be on maintaining confidentiality of sensitive information.

4.3 Software Requirements Specification for Security model for an object oriented design (SMOOD):

A software requirements specification (SRS) is a complete description of the behaviour of the system to be developed. The SRS contains functional, non-functional or supplementary requirements. Non functional requirements include requirements, which impose constraints on the design or implementation (S. L. Pfleeger, 2001).
4.3.1 **Security model for an object oriented design (SMOOD):**

In case of object oriented software, the design phase was most appropriate phase to estimate security since the quality of object oriented software was highly dependent on its design and security was one of the attribute of quality. In the propose work, an attempt was made to combine all important research finding and prepared base for our research work. In addition to that the development of a security model (SMOOD), which could estimate security of design. With the help of proposed security model designer can verify the security of design. If several designs were presented then designer could definitely select the secure design or if the designer discovers that security of present design was less. Then it is essential to do required alteration in the design stage.

4.3.2 **Purpose:**

The purpose of Software Requirements Specification (SRS) document was to describe the proposed security model. Essential consideration related to the development of the security model. SRS describes other factors necessary to provide a complete and comprehensive description of the requirements for the software. SRS captures the complete software requirements for the system, or a portion of the system. Requirements described in requirement gathering were elaborated and provide details for developing security model for object oriented design (SMOOD).

4.3.3 **Scope:**

The Scope of this research work was associated with the design properties of object oriented methodologies, which were unswervingly evaluated through analysing the internal as well as external assembly, association and functionality of methods, attributes, design components and classes. Valuation of a class definition by specifying its attributes and methods and relationship with other classes unveils noteworthy information that neutrally captures the functional and structural features of a class and its objects. Some design properties were same for structural and object oriented development like abstraction, encapsulation, coupling, cohesion, complexity and design size, and some additional design properties for object oriented development like: inheritance, messaging, polymorphism, composition as well as class hierarchies. All these properties were considered in this research work.
A class diagram is used as input to the security model (SMOOD). A class diagram not only documents, envisages and describes various aspects of a system but also builds executable code of the software application. The class diagram showcases a collection of classes, associations, interfaces, constraints and collaborations. It is also called a structural diagram. The class diagram was used to analyze and design the static view of an application. It describes duties of a system, base for component and deployment diagrams, forward and reverse engineering.

4.3.4 Overall Description:
Product perspective: The proposed SMOOD model gives guideline to the designer to map the requirement in such a way that it ultimately produce secure design. The secure design was a design with no defect. If software was protected by design then it’s very rigid for impostors or external intrusions to breakdown the software.

4.3.5 Features:
The features that were described in this SRS document were used in the future phases for development of security model. The features will help all the developer for the essential development from initial level to complete development. The features described here meet the needs of all the users. The success criteria for the system were based in the level up to which the features described in this document were implemented in the system. The system consists of the following features:

- **Security model calculated on class diagram:**
  In literature many of quality model and security metric are based on class diagram. The class diagram showcases a collection of classes, associations, interfaces, constraints and collaborations. It is also called as structural diagram. The class diagram was used to analyze and design the static view of an application. In the class diagram, based on design properties and confidential data security model were calculated.

- **Security model should be estimate security in quantitative manner.**
  Many qualitative models were present for security in literature but they were unable to give the proper security measurement. It was stated by many researchers that for practical purpose security must be estimated in quantitative manner. The proposed model was based on design properties so it should give quantitative measures.
• **Security model should be implemented from bottom up level:**
  This approach was not covering the application level security or emerging security polices in software life cycle. Focus was only on the design of software, testing the security of design and providing necessary data for designer to develop secure design. Focus was on relating the bottom level details with high level component.

• **Special attention need to give for confidential information:**
  The sensitive information, special attention needs to be given in the designing phase for keeping the information confidential. To achieve this goal, detailed knowledge of the causes and prevention of vulnerabilities were essential.

4.3.6 **Modules:**
The modules that were described in this SRS document can be used in the future phases for development of security model. The system consists of the following modules:

• **Elicitation of important design properties that influence security:**
  All the design properties of object oriented methodologies were considered at the initial phase. After that each design property was considered as security point of view. Finally those design properties which influence the security were considered for further development.

• **Finding the influence of each design properties on security:**
  Once all the design properties related to the security were listed then the influence of each property was examined for the positive and negative influence. Once the influence was considered then the range of that influence was determined (either high or low).

• **Elicitation of important security attributes.**
  Security was expressed in terms of the security attributes and it can measure in terms of security attributes.

• **Finding relation between design properties and security attributes:**
  Software security was combination of internal attributes and external attributes. Internal attributes includes design properties and external attributes consist of security attributes. To find the relationship between these two security principals were considered.
• **SMOOD model development:**
  Based on the relationship between the security attributes and object oriented design properties and the influence that each design property having on security the security model was developed.

4.3.7 **Assumptions and dependencies:**
- The security model was developed for object oriented methodology.
- Design properties can be calculated from class diagram.
- All the confidential data was required to identified and specify in the diagram.
- Confidential data need to mark with UML sec notation.

4.4 **Requirements Validation:**
The goal of requirements validation was to seek out and correct problems before resources were committed to implementing the requirements. It was concerned with examining the requirements to certify that they meet the stakeholder’s intentions, and to ensure that they define the right system, the essence of the agreement and understandings between developer and acquirer about what to build, in a manner that ensures a common understanding across the project team and among the stakeholders. Validation addresses specific requirements and the requirements collection as a whole. For the proposed SMOOD system the requirements reviews were conducted to validate that requirements may be correct, unambiguous, complete, consistent, ranked for importance, verifiable (testable), modifiable, and traceable. Review was conducted between the developers and the guide.

4.5 **Discussion:**
The requirements were given by the customer or the client. The customer requirement document was reviewed and the changes were made in the ‘acquisition processes’. The requirements were now explained in detail and made more understandable such that the designers might aware of the requirement more in depth and have a clear idea about what they could design. The requirement document called as the software requirement specification (SRS) was prepared. The requirement document was as soon as created the review process should be made. It was good to find out the faults
and errors or bugs in early stages of the SDLC since the cost increases exponentially throughout the SDLC.