MESR
منار الشرق للدراسات والأبحاث
Manar Elsharq for Studies & Research
Manar Alsharq Journal
for Education and Instructional Technology
Homepage:
http://meijournals.com/ojs/index.php/majeit/index
ISSN: 2790-6698
مجلة منار الشرق
للتربية و تكنولوجيا
التعليم

تصميم نموذج هندسة المتطلبات لتحقيق جودة البرمجيات

Designing a Requirements Engineering Model to Achieve Software Quality

محمد علي بدوي العوض كنين

Mohammed Ali Badawy Alawad Kenen

استاذ مساعد. جامعه وادي النيل. كليه علوم الحاسوب وتقانه المعلومات

Assistant Professor, Nile Valley University, Faculty of Computer Science and Information Technology[1]

استلام البحث: 20/02/2026 **مراجعة البحث:** 18/03/2026 **قبول البحث:**17/04/2026

الملخص:

من اهم المشاكل التي تناولتها الدراسة ان هنالك غموض في فهم المتطلبات التي تخص المشاريع البرمجية بصورة جيدة وكثير من المستخدمين من العملاء لا يدركون ماذا يريدون من النظام او الغاية من بناء النظام وهذا طبعا يتسبب في إعاقة نجاح المشاريع البرمجية وكان من أهم اهداف الباحث في هذه الدراسة الوصول الى كتابة المتطلبات بصورة عملية وفق معايير و نموذج المقترح من قبل الباحث وهو عبارة عن مجموعه خطوات مرتبة منظمة سوف تساعد في كتابة المتطلبات بصورة صحيحة ومن اهم النتائج التي توصل اليها الدراسة بعد طرح عدد من الاستبيانات على عدد من الشركات البرمجية والافراد ساهم الاستبيان في فهم وتحليل واقع هندسة المتطلبات داخل المؤسسات البرمجية ومدى استخدامه لمعايير عملية لقد وضح الاستبيان حقائق كثيرة جدا واهمها عدم استخدام المؤسسات البرمجية وحتى الافراد في اتباع نموذج معين لجمع المتطلبات.

الكلمات المفتاحية: هندسة البرمجيات – هندسة المتطلبات – هندسة النظم – التحقق من جودة البرمجيات – النموذج – المتطلبات الوظيفية – المتطلبات غير الوظيفية – دراسة الجدوى – المتطلبات الوظيفية – استخلاص المتطلبات – وثيقة متطلبات منهجية أجايل – البيانات

Abstract

One of the key issues addressed in this study is the ambiguity surrounding the understanding of software project requirements. Many users and clients often lack a clear vision of what they expect from the system or the purpose behind its development, which inevitably hinders the success of software projects. A primary objective of the researcher in this study was to establish a practical approach for writing requirements based on specific standards and a proposed model. This model consists of a structured and organized sequence of steps designed to facilitate accurate and effective requirement specification. Among the most significant findings of the study—based on a series of questionnaires distributed to various software companies and individuals—was the contribution of the survey in analyzing the current state of requirements engineering within software institutions and assessing the extent to which practical standards are applied. The survey revealed numerous insights, most notably the widespread absence of a standardized model for requirements gathering among both organizations and individual practitioners.

Keywords: Software Engineering - Requirements Engineering - Systems Engineering -Software Quality Verification - Model - Functional Requirements - Non-Functional Requirements - Feasibility Study - Functional Requirements - Requirements Elicitation - Agile Requirements Document - DATA

Introduction

Computer software has become an essential part of our daily lives, governing both simple and complex activities. Software now controls aircraft, nuclear reactors, spacecraft, surgical procedures, banking transactions, telecommunications, and commercial operations of all kinds. It also operates numerous electronic devices used by individuals even at the household level, such as electric ovens, vacuum cleaners, and more.

Several major countries have come to rely heavily on the software industry as a cornerstone of their economies, investing vast sums in the development of high-quality software that meets user needs. Given the critical importance of software projects, any error within them can lead to catastrophic human and financial consequences. Studies have shown that over 85% of software project failures are due to a misunderstanding of software requirements—that is, a lack of clarity regarding the actual needs and expectations of stakeholders. Poorly defined or ambiguous requirements can result in the failure to achieve project goals, increased maintenance costs, and delays in delivery schedules. Most problems and failures in software projects stem from an inadequate understanding of the problem to be solved—the very purpose of the software itself.

In the early 1960s, the foundations of software engineering were laid to address the recurring issues and errors in software projects. Researchers in the field introduced a new scientific domain known as Requirements Engineering, which is considered the cornerstone of software projects and the initial phase in software product development. This phase involves planning, feasibility studies, and analysis of the project's surrounding environment. The final output of this phase is the Software Requirements Specification (SRS) document, which must contain accurate, complete, consistent, and well-structured requirements. The SRS serves as the contractual basis between stakeholders and developers, as well as a reference guide for all project participants—including developers and end users. It is regarded as the foundation for the design phase and all subsequent stages of the software project. The more precisely and accurately a software engineer can define the requirements, the closer the project moves toward successful completion.

The fundamental measure of success for any software project is the extent to which it fulfills the purpose for which it was originally designed. The requirements phase is defined as the stage dedicated to discovering the project's intended goals from the perspective of managers and stakeholders Software projects rely heavily on a clear and comprehensive understanding of the problem to be solved, as well as an accurate grasp of user needs and expectations—what the system should and should not be able to do. These expectations and requirements must align with the overarching objectives of the organization benefiting from the software project Quality attributes are themselves considered types of requirements, such as maintainability, portability, and reliability. Within the Software Requirements Specification (SRS) document, it is essential to describe system functions, performance criteria, design constraints, and quality characteristics. All of these requirements must be defined in a way that enables their successful implementation In conclusion, well-defined, precise, and clear requirements can lead to the development of a high-quality software product. They have a positive impact on both process and product quality, facilitating correct and efficient development, fulfilling the intended purpose of the software, reducing development time, and supporting task execution. In contrast, poorly defined or ambiguous requirements can negatively affect software quality—both in terms of process and product. They may lead to increased complexity, failure to achieve the software's intended goals, difficulty in predicting development outcomes, and flawed implementation, ultimately resulting in higher long-term costs for development and maintenance.

Requirements Engineering

refers to the process of identifying, documenting, and maintaining requirements within the context of engineering design. It provides the appropriate mechanisms for understanding what the client wants, analyzing their needs, evaluating economic feasibility, negotiating logical solutions, clearly defining the proposed solution, verifying specifications, and managing requirements throughout their transformation into a functional system. Therefore, Requirements Engineering is the disciplined application of proven principles, tools, and notations used to describe the intended behavior of the proposed system and the constraints associated with it.

Study Problems

The primary issue addressed in this paper is the ambiguity surrounding the understanding of software project requirements. A significant number of clients and software users are often unaware of what they truly need from the system or the purpose behind its development. This lack of clarity regarding actual requirements and stakeholder needs poses a serious obstacle to the success of software projects. It prevents the achievement of intended goals, threatens financial budgets and project timelines, and leads to confusion about the tasks to be executed.

From this central problem, several critical sub-issues emerge, including:

1. **Inability to manage requirements effectively** within software projects, which has a major impact—especially when sudden changes in project requirements occur.
2. **Lack of understanding of both functional and non-functional** requirements within the project, resulting in incomplete or misaligned system specifications.
3. **Insufficient comprehension of the requirements lifecycle**, which affects planning, tracking, and adapting requirements throughout the development process.

The main problem addressed in this research paper can be summarized as the lack of clarity in understanding software project requirements. A considerable number of clients and software users are often unaware of what they truly want from the system or the actual purpose behind its development. This ambiguity regarding real needs and requirements significantly hinders the success of software projects, prevents the achievement of intended goals, threatens financial budgets and project timelines, and leads to confusion in task execution. From this core issue, several related problems emerge, the most critical of which include:

Importance of the Study

- This research addresses one of the most critical phases in software product development: the requirements phase. It focuses on defining the objectives of this stage and clarifying its integration both within the engineering process itself and in terms of the final output of requirements engineering.
- The requirements definition phase is considered the most essential stage in building software systems. Requirements must be specified with precision, as all subsequent development phases rely entirely on them.
- There is a pressing need to evaluate the quality of software performance in relation to requirements engineering, in order to identify the root causes of system failures.

Study Objectives

The objectives of this research can be summarized as follows:

1. **To examine the current state of requirements engineering** in a selection of software development companies in Sudan, and to identify the key challenges that hinder the application of standards in defining software requirements. The study also aims to explore the methods currently used for writing requirements and assess their alignment with scientific methodologies.
2. **To investigate the relationship between software requirements and quality factors** in the context of software development.
3. **To design a model for the requirements engineering process** that enables the formulation of clear, complete, and accurate requirements which reflect the true purpose of the software. The goal of the proposed model is to minimize software-related failures and enhance the quality of the final product.
4. **To evaluate the proposed model** through feedback from experts working in software companies.

The primary objective addressed in this research paper is the design of a model that enhances the understanding of software project requirements. A well-structured model will assist clients and software users in recognizing what they truly need from the system, the purpose behind its development, and their actual requirements. Below are some of the secondary objectives:

Study Methodology

This research paper is structured around several key components:

1. **Theoretical Framework**: This section explores the fundamental concepts of requirements engineering and software engineering. It covers all aspects related to requirements, including their types, elicitation techniques, commonly used models, software quality attributes, and methods for measuring quality.
2. **Informational Framework**: This part involves collecting data related to the current state of requirements engineering within the targeted software companies. Data collection is conducted through questionnaires, and the descriptive approach is adopted for this component.
3. **Analytical Framework**: In this section, the study analyzes the reality of requirements engineering in the selected companies by examining the data gathered through the questionnaire. The analysis is performed using the Statistical Package for the Social Sciences (SPSS).
4. **Evaluation and Conclusion**: This final component focuses on evaluating the results and proposing recommendations to overcome the challenges faced by companies in defining and documenting software

requirements. The evaluation is based on established standards and scientific models, leading to a set of practical suggestions.

Scope of the Study

This study aims to design a practical model for requirements engineering that contributes to enhancing software quality by addressing issues related to the misinterpretation and lack of clarity in understanding requirements among clients and users. The scope of the paper focuses on analyzing the current state of requirements engineering within software development environments and assessing the extent to which scientific standards are applied in writing and documenting requirements.

A questionnaire was distributed to a group of software companies and professionals in the field of software development in Sudan, with the goal of collecting both quantitative and qualitative data. This data supports a deeper understanding of the challenges organizations face in accurately defining requirements and the impact of these challenges on the quality of the final software product.

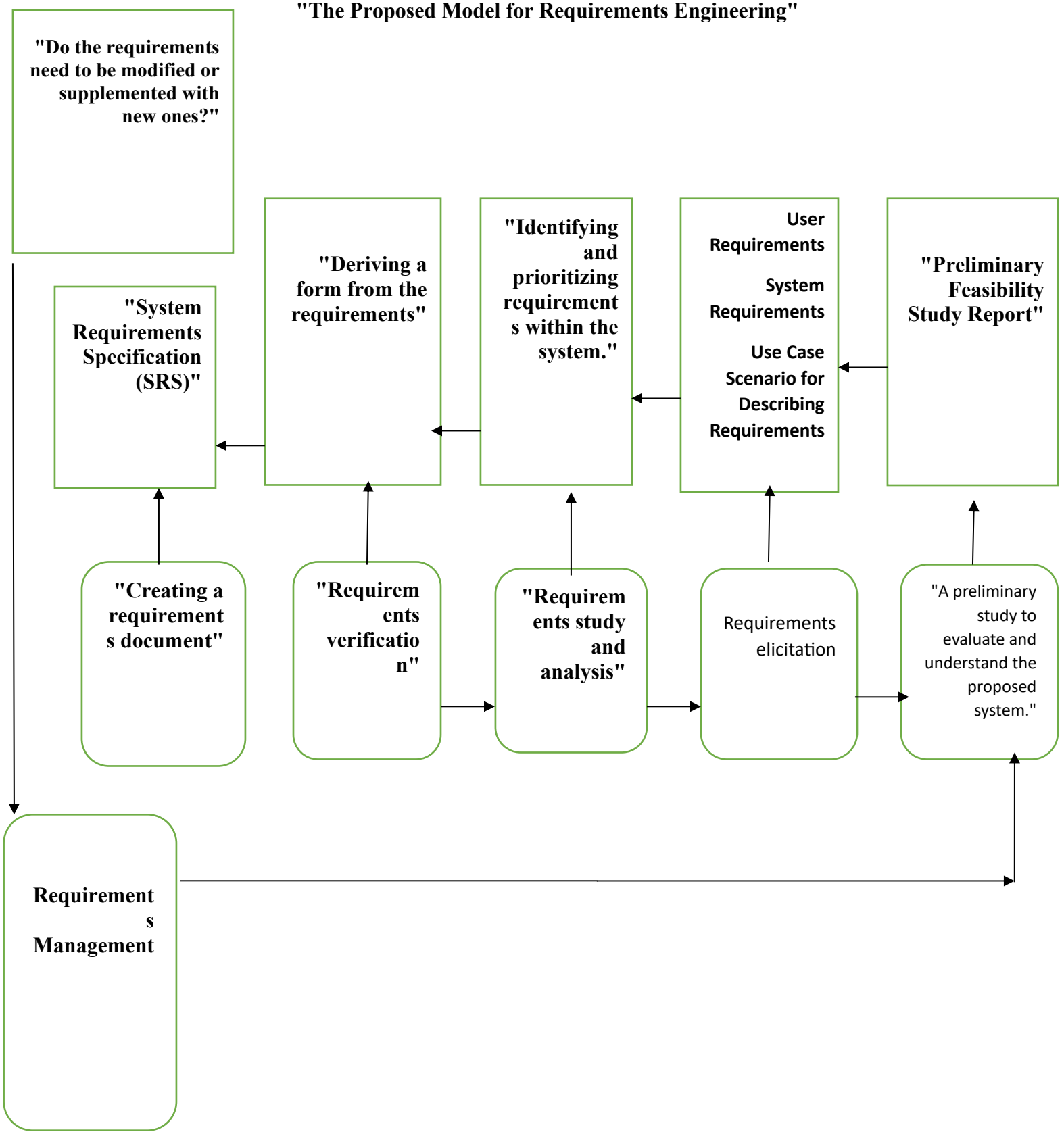
The study covers the following aspects

- ✓ Analyzing the relationship between requirement quality and software quality.
- ✓ Examining current practices used in gathering and documenting requirements.
- ✓ Designing a structured engineering model that facilitates clear and complete requirement specification.
- ✓ Evaluating the proposed model based on feedback from experts and practitioners in the field.

The scope of the study is centered on software projects within Sudanese companies, with the potential for the proposed model to be generalized and applied to similar development environments, paving the way for broader future applications.

"The Proposed Model for Requirements Engineering"

"The Proposed Model for Requirements Engineering"



Results and Recommendations

Study Results

Based on the analysis of the sample of individuals who received the questionnaire, the researcher reached the following findings:

✓ A number of software companies do not conduct preliminary studies for their software projects.
✓
✓ The researcher found that a certain number of respondents (4) apply requirements elicitation techniques.
✓ The study revealed that several participants (7) do not use scientific methods in software project development.
✓ The results showed that some respondents (4) are unfamiliar with Agile modeling.
✓ It was also found that some software companies do not rely on documenting software requirements through formal specification documents.
✓ A large number of companies place strong emphasis on reviewing requirements.
✓ The results indicated that many companies and individuals lack understanding of both functional and non-functional requirements.
✓ The researcher found that companies and individuals commonly use interviews as a method for gathering requirements.
✓ The surveyed software companies showed little concern for adhering to the scheduled delivery timelines of software projects.

Recommendations

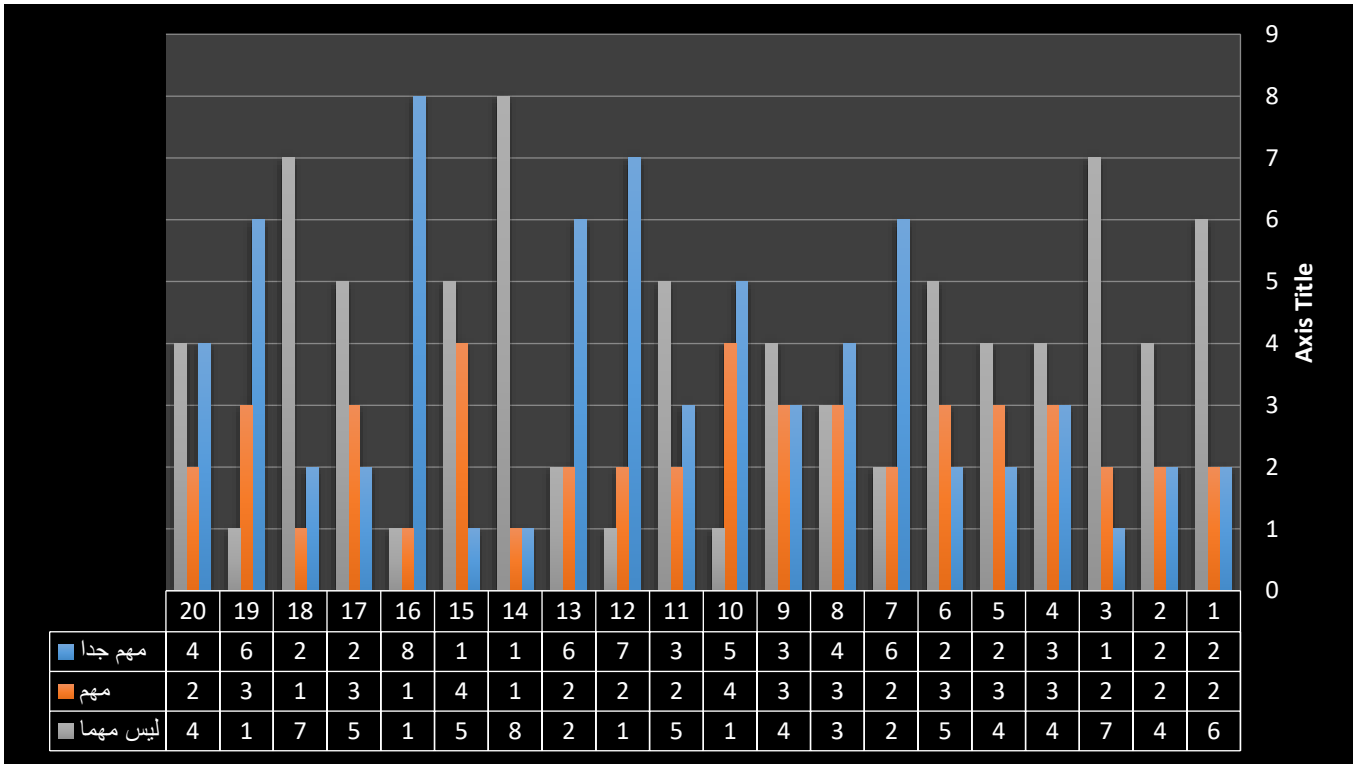
✓ Adhere to scientific standards in the process of building software systems in alignment with requirements engineering principles.
✓ Utilize advanced analysis techniques such as requirements modeling and early validation to ensure clarity and accuracy of requirements.
✓ Integrate quality standards from the requirements gathering phase—not only during testing or implementation.
✓ Strengthen collaboration between analysis, design, and quality assurance teams to ensure that requirements align with quality objectives.
✓ Conduct regular reviews of project outputs and update the model based on feedback.
✓ Apply change impact analysis tools to assess how modifications to requirements affect the quality of the final product.
✓ When documenting software requirements, follow the standardized format of the Software Requirements Specification (SRS) as defined by IEEE.

Requirements Engineering Questionnaire

Very Important	Important	Not Important	Questionnaire Item	No.
2	2	6	Do you conduct feasibility studies when initiating projects?	1
2	2	4	Do you apply requirements elicitation in project development?	2
1	2	7	Does the company conduct preliminary studies to understand unclear requirements?	3
3	3	4	Do software companies use practical methods for requirements elicitation?	4
2	3	4	Do companies meet with all stakeholders during the requirements elicitation phase?	5
2	3	5	Does the company use a standardized format for the requirements document?	6
6	2	2	Does the company distinguish between functional and non-functional requirements?	7
4	3	3	Does the company use a standardized and editable format for the requirements document?	8
3	3	4	Does the company follow a specific scientific methodology?	9
5	4	1	Does the company use the Agile methodology?	10
3	2	5	Does the company prioritize reviewing requirements?	11
7	2	1	Does the company involve all stakeholders during the requirements phase?	12
6	2	2	Does the company use an editable requirements document?	13
1	1	8	Does the company use interviews to gather requirements?	14
1	4	5	Does the company use questionnaires to gather requirements?	15
8	1	1	Does the company use observation as a method for gathering information?	16
2	3	5	Does the company use all available data collection methods?	17
2	1	7	Does the company work as a unified team in reviewing each project phase?	18
6	3	1	Does the company consult experts in requirements engineering?	19
4	2	4	Does the company adhere to a specific timeline for requirements engineering?	20

Note

- ✓ **Not Important**: Indicates that the software company does not prioritize the corresponding item.
- ✓ **Important**: Indicates that the company considers the item relevant.
- ✓ **Very Important**: Indicates that the company strongly prioritizes the item.



	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
مهم جدا	4	6	2	2	8	1	1	6	7	3	5	3	4	6	2	2	3	1	2	2
مهم	2	3	1	3	1	4	1	2	2	2	4	3	3	2	3	3	3	2	2	2
ليس مهما	4	1	7	5	1	5	8	2	1	5	1	4	3	2	5	4	4	7	4	6

"A graphical representation of the survey findings from ten software firms."

References and Sources

- ✓ 1-عبد الحميد بسيونى.2005.مبادئ هندسة البرمجيات.دار الكتب العلمية للنشر والتوزيع.
- ✓ 2-عبد الحميد بسيونى.2005.أساسيات هندسة البرمجيات.دار الكتب العلمية للنشر والتوزيع
- ✓ 3- .زاهر حسين الحاج.2006.هندسة البرمجيات ثنائية الهندسة والإدارة.شعاع للنشر والعلوم
- ✓ 4- ورقة بحثية – جامعة تشرين: تطوير النموذج الشلالي باستخدام منهجيةCRISP-DM
- ✓ Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education.
- ✓ Kotonya, G., & Sommerville, I. (1998). Requirements Engineering: Processes and Techniques. Wiley.
- ✓ Pressman, R. S., & Maxim, B. R. (2014). Software Engineering: A Practitioner's Approach (8th ed.). McGraw-Hill Education.
- ✓ Glinz, M. (2007). On Non-Functional Requirements. In International Conference on Requirements Engineering (pp. 1–7). IEEE.

- ✓ Mishra, A., & Dubey, A. (2020). A Requirements Engineering Process Model for Software Development. International Research Journal of Engineering and Technology (IRJET), 7(7), 1234–1240.
- ✓ Nuseibeh, B., & Easterbrook, S. (2000). Requirements Engineering: A Roadmap. In Proceedings of the Conference on the Future of Software Engineering (pp. 35–46). ACM.

Online Sources:

- ✓ huzam.net/ARABIC/html. Last access: june. 27, 2015.
- ✓ ceur-ws.org/Vol-69/paper10.pdf.Last access: july. 11, 2015.
- ✓ http://www.ijric.org/Volumes/Vol2/6Vpol2.pdf. Last access: july. 18, 2015.
- ✓ http://www.wikipediak.org/wiki. Last access: july. 22, 2012.
- ✓ http://www.aymansultan.com. Last access: ogust. 2, 2015 .
- ✓ http://www.faculty.ksu.sa/ahafez/documents/days%201-1.ppt.Last access: ogust. 13, 2015.