Vol.4, No.5, Page 2139- 2165, 2026 https://doi.org/10.59628/jast.v4i5.2201 ISSN 2958-9568 JAST

SUN PUBLISHER

مجلة جامعة صنعاء للعلوم التطبيقية والتكنولوجيا

Sana'a University Journal of Applied Sciences and Technology

https://journals.su.edu.ye/index.php/jast/

Factors Influencing the Acceptance and Adoption of Open-Source Software: A Concept-Centric Systematic Literature Review

Sumaia A. Al-Edresi [1*] and Adnan Y. Al-Mutawakkil[2]

[1]**Department of Information Technology, Faculty of Computer and Information Technology, Sana'a University, Sana'a, Yemen,**
[2]**Department of Computer Science, Faculty of Computer Sciences and IT, Sana'a University, Sana'a, Yemen.**

*Corresponding author: smia.aladrisi@su.edu.ye

Abstract

Free and open-source software (FOSS) is software in which anyone is freely licensed to use, copy, study, and change the software in any way. The source code is publicly shared, encouraging people to improve the software design voluntarily. This is in contrast to proprietary software, where the software is subject to a restrictive copyright license and the source code is usually hidden from users. Many countries are witnessing a significant transition towards adopting open-source software within their government structures, coupled with efforts to develop strategies and policies that support this trend. This change is attributed to several reasons, most notably savings in the total cost of ownership, freedom to copy and distribute software, compliance with legal controls, high reliability, a variety of available programs, and rapid performance, in addition to the multiple educational and administrative benefits. Despite this remarkable expansion, it remains necessary to understand the potential of this software for individuals and entities while identifying the factors influencing its adoption. Accordingly, this study aims to explore the acceptance of open-source software and analyze the technical, administrative, and personal factors that promote its use, especially in environments undergoing a transition towards sustainable digital systems. The study collected several papers published between 2007 and 2023, approximately 85 different research studies combining theoretical and experimental aspects. After that, 61 papers were selected from among them and analyzed. The results revealed six main categories that influence the acceptance and adoption of open-source software. The study recommends developing a comprehensive model that integrates all behavioral, social, cultural, and technical dimensions to identify the factors influencing the acceptance and adoption of open-source software. The study also emphasizes the need for future empirical research in the Yemeni context, especially in public and educational sectors, to validate the integrated framework, and provide evidence-based policy recommendations for effective OSS adoption.

Article Info

Keywords:
adoption, open source, acceptance, usability, factors, influencing.

Article History:
Received: 26-September-2025,
Revised: 24-October-2025,
Accepted: 28-March-2026,
Published: 28 May 2026.

1. INTRODUCTION

There are several different applications of the term "open source." The most common occurrence of software is a product of the software revolution of the 1960s, which introduced the concepts of "real-time" operations, operating systems, and computer games. As the user-friendliness of software increased, computer games and word processors were developed. These products did not improve our computers but guaranteed an ethos of proprietary (closed) software.

Conflict of Interest declaration: The authors declare that they have no affiliations with or involvement in any organization or entity with any financial interest in the subject matter or materials discussed in this manuscript.

Author Contributions: AB and MJ contributed to the design and implementation of the research, JK to the analysis of the results and to the writing of the manuscript. VK conceived the original and supervised the project.

Open-source software is often not free, but its defin-

ing characteristic is free/open-source copy/light software. The movement towards open source was clouded by propaganda against the feasibility of privately developed "pragmatic" software and its credibility. The birth of the open-source movement (1998) and the foundation of the well-known organization “Open-Source Initiative" (OSI) strengthened free code programs and encouraged the development of operating systems. The Open-Source Initiative" defines open-source software through the open-source logical agreement, motivated by the need to develop software whose objects and sources can be freely conferred to the public.

Building on this foundation, Open-Source Software (OSS) is a type of software where the source code is available and can be seen by everyone. This makes it easier for others to use, change, and build upon the software in the future, enhancing and distributing it. Many companies and professional sectors are aware of the benefits of Open-Source Software (OSS), which include reduced development costs, a quicker development cycle, and availability of source code.

The fundamental principle of open-source software is the unrestricted ability to view, alter, repurpose, and share the source code, regardless of expense. Individuals and organizations can adopt Open-Source Software (OSS) in various ways, such as integrating OSS components with existing systems or implementing OSS in their environment as end users. Numerous studies concur that the absence of license costs associated with free open-source software (FOSS) facilitates quicker technology adoption and speeds up the innovation ecosystem, claiming that the use of FOSS might lead to increased supplier independence and contract freedom, as well as the development of a local software sector [1].

Advantages of Choosing Open-Source Systems

Advantages of choosing Open-Source Systems are more significant because of [2], [3]:

- Better adaptability: Because the software source code is available and can be freely modified, customization is usually easier.
- Reducing resource dependence: When company relies on a proprietary solution, it becomes tied to the company that owns the project. If that company decides to stop supporting the project or exit the market, this could pose a significant threat to the continued updating and maintenance of the user company’s ERP system, since it will not have access to the source code of the software.
- Lower costs: Open-source ERP systems do not require paying for licenses and often don’t need expensive hardware to work well.
- Use of standard hardware.
- More flexibility with fewer restrictions from vendors.
- No need for specialized consultants or tools for setup.
- Higher productivity due to faster learning curve.
- A large community of experts.
- Use of free and open-source tools like databases and operating systems.
- Ability to change the source code to fit the company’s specific needs.

A recent Gartner survey indicates that the amount of private source code in companies’ software has decreased over time. Many businesses now depend almost completely on open-source technologies. Software such as the Linux operating system or the Apache HTTP Server has become a standard, and it has helped create new products and services in areas such as the Internet of Things (IoT), big data, and cloud computing. The growth of open-source software has also been aided by more companies and organizations supporting it, such as GitHub. Com and Red Hat Inc. In addition to private businesses recognizing the value of open source, the government sector has also started to use open-source ideas and work with open-source communities, for example, by including them in how they buy software and services [4].

In developing countries, using open-source software can help the economy grow. This is mainly because of the open-source

movement’s belief in independence, which is against paying for licenses and buying software. Many governments have made it part of their national technology plans and programs [5]. However, even though free and open-source software (FOSS) is commonly used in developed countries such as Europe and North America, it has not been a steady or long-term success in most public organizations in developing countries [1].

The use of free and open-source software (FOSS) in schools is becoming increasingly common. Many countries are starting to require the use of FOSS in their government offices, while others are working on creating policies for this software. This growing trend is due to several reasons, such as lower overall costs, the freedom to copy and share software, staying within legal rules, dependable performance, easy access to software, efficient operation, strong security, and many helpful benefits for education and administration [6].

Universities and educational institutions rely on several software programs and tools to support teaching and training processes. Despite the variety of options currently available on the market, most are expensive and not cost-effective. Choosing the right tool for academic work can be difficult because these tools are often expensive. In recent years, an increasing number of people have started using free and open-source software. Some countries, such as Russia, require these tools in

all educational institutions. Other countries are slowly moving towards their use. The level of adoption varies between countries and organizations, but the fundamental reason for this trend is cost reduction. For example, colleges and universities in the United States often find that using open-source software, such as Moodle, is both more flexible and cost-effective [6].

Although open-source software (OSS) offers numerous advantages, its mainstream adoption is beset by unique obstacles. One such issue is the lack of appropriate documentation; for example, the progression process involves programmers from different societies and locations worldwide who do not document the code suitably. Usability, configuration, and design issues also exist [7].

However, the operation of OSS often requires special knowledge and qualifications that are not provided by learning institutions. Thus, its adoption is stymied among intrinsically motivated adopters. There is also a need to reinforce the generic skills of visibility and training, which significantly influence the adoption of open-source software (OSS).

There is a need to create usable OSS due to the expanding due to the expanding number of people who are not developers but use open-source software (OSS), along with the use of OSS applications. Usability is a property linked to software sustainability and is a crucial quality aspect that must be considered. Software usability is an important factor that is measured based on how well it works, how easy it is to use, and how satisfied users are with it in a specific situation (ISO, 2018). It is also a key part of models that explain how people accept new technology, such as the model by Venkatesh et al. (2003) and the information success model [1].

The amount of OSS used by businesses around the world is different. Many factors influence whether people accept OSS, such as the software itself, social factors, and infrastructure, there are not many studies looking at how enterprise users accept OSS [8].

This study contributes to the growing knowledge base on open-source software (OSS) acceptance and adoption by providing a comprehensive systematic review of existing research. Previous research has focused on specific user groups or theoretical frameworks, whereas this study synthesizes and analyzes the results of 61 peer-reviewed studies covering diverse fields such as education, development environments, and institutional use. This study is distinguished by its organization of findings according to clear conceptual themes rather than presenting factors in isolation, which facilitates understanding how technical, organizational, and psychological factors interact to shape open-source software acceptance and adoption.

In addition, this study analyzes and compares three major acceptance models—TAM, UTAUT, and SDT— in the context of OSS adoption. To the best of our knowledge, these models have rarely been examined together in the context of OSS, which provides this review with a more integrated perspective. Finally, by summarizing where the literature converges and where gaps remain—such as the limited attention to cultural or motivational factors—the review can help guide future research efforts in more focused and meaningful directions.

This study aims to answer the following research questions:

· **RQ1:** What are the key factors and theoretical models that influence the acceptance of Open-Source Software (OSS)?

· **RQ2:** What are the main barriers that impede the acceptance of OSS?

· **RQ3:** What integrated frameworks and policy interventions can be proposed to enhance OSS acceptance, particularly in developing countries such as Yemen?

The remainder of this paper is organized as follows: Section 2 provides a systematic review of the related literature. Section 3 summarizes the methodology used to select and analyze the studies. Section 4 examines the key factors influencing OSS acceptance, and Section 5 analyzes the theoretical models (TAM, UTAUT, and SDT) applied in this context. Section 6 identifies the technical, organizational, and policy barriers to OSS adoption. Finally, Section 7 synthesizes the findings and discusses the implications for the Yemeni context, followed by the conclusion and future research directions in Section 8.

2. LITERATURE REVIEW

Understanding how people accept and use software systems is a critical issue in software studies. Many researchers have studied this field from different perspectives to identify the factors that influence a user's decision to adopt a particular software.

Silva et al. [1] A study was conducted to examine the factors that affect the desire to use free and open-source software (FOSS) in developing countries. This study sought to understand how several factors—such as low price, system effectiveness, ease of use, integration with other systems, and security—affect individuals' intention to use this software. The results showed that the most important factors influencing the decision to adopt open-source software were cost reduction, performance expectations, system quality, community influence, compatibility, and ease of use, whereas expected effort had no significant effect.

Ebardo [5] studied how individuals interact with open-source software (OSS) in terms of perception, acceptance, and use at the individual level. In one applied study, the model was used to analyze the effect of social, personal, and environmental factors on the adoption of OSS technologies by users in a developing country. Ebardo also analyzed the role of the factors "training" and "visibility" in supporting students' adoption of open-source software. Additionally, this study helps bridge the

knowledge gap regarding the prevalence of open-source software use in developing nations. The results of the pilot study showed that users' access to training significantly enhanced their perception of the software's ease of use and usefulness. Furthermore, the availability of the software in a visually and visibly appealing form in the learning environment further enhanced perceptions of its ease of use. These results support the findings of previous studies, while others indicate challenges in implementing OSS within organizations. Accordingly, universities should improve the quality of teaching content and develop policies that encourage the use of OSS within the academic environment.

Although both Silva et al. and Ebardo's studies address developing country settings, their focus differs; Silva highlights structural factors such as cost and integrability, while Ebardo focuses on individual aspects, such as the impact of training and the visibility of software within the educational environment.

Al-Hajri et al. [6] addressed the use of free and open-source software (FOSS) in the IT program at the College of Applied Sciences in the Sultanate of Oman. To achieve this, the researchers adopted an alternative educational framework to identify appropriate open tools for the major's courses, focusing on key elements such as technical planning, the formation of follow-up committees, training, awareness raising, policy development, and stakeholder engagement. The results indicate a clear increase in the adoption of open-source software within educational institutions worldwide, with many countries beginning to enforce its use in the government sector. This phenomenon is driven by several factors, most notably low ownership costs, free distribution, compliance with regulations, credibility, ease of use, effective performance, and security, in addition to numerous educational and administrative advantages. The study concluded that leveraging open-source software offers institutions clear advantages, such as reduced software costs, access to the source code, and facilitating content exchange without licensing restrictions. Based on this, the researchers recommended the formation of specialized committees in each academic field to evaluate curriculum outcomes and modify them to suit the available open-source software.

Alrawashdeh et al. [8] addressed the issue of software acceptance, which has received considerable attention in previous research. Regarding open-source software, the team noted the paucity of studies examining how individuals and organizations adopt it.

For example, Guibo and Wang (2011) presented a model based on the Technology Acceptance Model (TAM) to explore the influence of social identity and personal creativity on users' willingness to use open-source software. A survey of 280 experts showed that the perceived usefulness of the software was not significantly more influential than the perceived ease of use, suggesting that open-source software may be complex or require effort to master. The results showed no significant relationship between perceived usefulness and personal creativity, whereas there was a positive relationship between personal creativity and software use. Furthermore, they demonstrated that social identity significantly influenced usage, perceived usefulness, and ease of use. Alrawashdeh et al. analyzed the factors influencing open-source software acceptance within organizations and designed a comprehensive model that combined open-source software attributes, infrastructure factors, and components of the Unified Theory of Acceptance and Use of Technology (UTAUT) model to better understand adoption drivers. This study revealed the influence of additional important factors, such as cost, community influence, a supportive environment for use, and user trust. The results showed that software quality, compatibility, and security are critical factors in raising performance expectations, which, in turn, influence the willingness to use open-source software. This was based on a survey of participants from the public and private sectors. Notably, the study found a close relationship between software ease of use and performance expectations. Easy-to-use software increases the user's sense of usefulness. Furthermore, it was found that IT professionals are more likely to adopt open-source software given their ability to modify and improve it to meet their specific needs.

Together, these studies (Al-Hajri et al., Alrawashdeh et al.) point to the essential role played by institutions such as universities and government agencies in adopting open-source software, with differences in the approach taken. While Al-Hajri presents a practical application framework, Alrawashdeh adopts an integrated theoretical model that includes unique characteristics of open-source software.

Racero et al. [9] studied students' intention to use open-source software after receiving mandatory training. The researchers relied on a conceptual model inspired by self-determination theory and the technology acceptance model, which included elements such as autonomy, competence, and social relationships, along with perceptions of ease of use and usefulness, and the behavioral intention to use the software. The results showed that intrinsic motivations, such as a sense of autonomy and social connection, positively impacted the perception of usefulness and ease of use, which in turn influenced students' willingness to use open-source software. The researchers concluded that strengthening these intrinsic motivations is essential to motivating the adoption of open-source software as an alternative to commercial software, emphasizing that mandatory training contributes to strengthening these motivations and encouraging its continued use. They also indicated that the study has important implications for encouraging the adoption of open-source software while highlighting some of its shortcomings.

Dawood et al. [10] believe that ease of use directly impacts the end user, in that it influences both their acceptance of open-source software and its maintenance. Therefore, ease of use is a vital factor in the software's success. This study analyzed 51 studies on aspects related to the development, testing, and evaluation of usability in open-source software, dividing these aspects into four main axes: improving usability, analyzing and evaluating usability, software selection, and customization. The researchers recommend integrating user experience specialists in the earliest steps of the software development process. User-centered design should also be adopted, as well as enhancing communication between programmers and users. These measures are essential for improving usability. These recommendations emerged from studies showing that many popular open-source applications contain advanced features that are difficult to use. This is due to the lack of attention paid by programmers, which hinders users' ability to modify and maintain them. The study also emphasizes the importance of considering usability as a fundamental pillar of the development plan to ensure the production of scalable, open-source software. To gather user feedback and implement the necessary software improvements, developers must employ various evaluation methods. Overall, this study offers a thorough examination of usability research in open-source software and opens up broad avenues for future research.

In both contexts, Racero et al. and Dawood et al. emphasized the value of usability, but from contrasting perspectives. Racero focused on the behavioral factors that influence students, while Dawood focused on software design quality, highlighting the interaction between user perspectives and technical considerations in determining open-source software acceptance.

3. METHODOLOGY

This review systematically assesses concept-centered literature to rank the findings related to the real motivations for adopting open-source software, thus providing clarity on what influences individuals to adopt open-source solutions.

3.1. Search Strategy and Identification

A systematic search was conducted in databases such as IEEE Xplore, Google Scholar, and ScienceDirect, covering the period from 2007 to 2023, to track the development of this field over time. Precise search terms with logical operators, such as ('OSS' or 'open-source software') combined with ('adoption', or 'acceptance', or 'usability') and ('barriers', or 'factors', or 'challenges'). An initial set of **85** carefully selected documents formed the core analytical basis of the study.

3.2. Screening and Selection

The selection process followed a rigorous screening protocol:

1. **Identification: 85** documents were identified through database searching.
2. **Screening:** After reviewing the titles and abstracts, it became clear that **7** documents fell outside the scope of this assessment because they did not address the research goals.
3. **Eligibility: 78** completed research papers were reviewed according to the study inclusion criteria. At this stage, **17** manuscripts were excluded to maintain a high level of analytical focus. The primary motives for exclusion were a narrow focus on technical aspects or a lack of sufficient experimental and intellectual depth to support a rigorous systematic assessment of understanding and ease of application of the method.
4. **Included:** The selection process resulted in a final list of **61** research documents. These documents formed the basis of the thematic analysis, providing the necessary depth to address the research questions.

3.3. Inclusion and Exclusion Criteria

Inclusion: To ensure scientific rigor, the selection process focused on research published in peer-reviewed journals, conference papers, and dissertations. Studies that adhered to established acceptance models, such as the Technology Acceptance Model (TAM) and the Unified Theory of Acceptance and Use of Technology (UTAUT), were strategically considered. This ensured that the analysis was grounded in a robust conceptual framework rather than relying on anecdotal evidence alone. Furthermore, to maintain linguistic and textual consistency, the review was restricted to research published in English, allowing for a rigorous and uniform examination of international literature.

· **Exclusion:** Non-scientific and commercial reports were excluded, with preference given to peer-reviewed research with proven credibility. The review period was strategically set to be limited to publications from 2007 onwards to ensure the survival of studies relevant to the open software environment, with the exception of a major historical study (Study No. 38), which was added to provide background on the growth of this field.

· **Quality Consideration**: Instead of using a formal qualitative assessment method, the research underwent rigorous scrutiny, and only studies that provided factual or conceptual data related to the determinants and barriers to the dissemination of open-source software were

Table 1. summarizes the literature.

Key Findings	Methodology	Sample Characteristics	Theoretical Models	Author(s) & Year	Ref
Cost reduction, performance expectations, system quality, community influence, compatibility, and ease of use were all found to be critical to adoption. Expected effort had no significant effect on the intention to use FOSS.	A quantitative and empirical study.	The study was conducted in Angola, targeting individuals to understand their intentions toward FOSS in a developing country context.	The authors proposed an original, new adoption model rather than just using a standard one. A key contribution of this model is the inclusion of Interoperability as a core element alongside traditional factor.	Silva et al. (2023)	[1]
Ease of use is identified as the most vital factor for both initial acceptance and long-term software maintenance. Many popular OSS applications possess advanced features but fail because they are difficult to use, primarily due to programmers' lack of attention to user needs. The study also emphasizes the importance of considering usability as a fundamental pillar of the development plan to ensure the production of scalable open-source software.	Fuzzy Delphi method.	A broad dataset consisting of 51 research papers covering various popular OSS applications and their development environments.	User-Centered Design (UCD) and Usability Evaluation Frameworks. The study moves beyond simple acceptance models to focus on the Software Development Life Cycle (SDLC) and User Experience (UX) integration.	Dawood et al. (2021)	[11]

Key Findings	Methodology	Sample Characteristics	Theoretical Models	Author(s) & Year	Ref
Found that performance expectancy and OSS characteristics significantly impact user acceptance. Software quality, compatibility, and security are critical in raising performance expectations. Ease of use has a direct positive relationship with performance expectations (perceived usefulness). Cost, community influence, supportive environment, and user trust influence adoption. IT specialists show higher adoption rates due to source code modifiability and customization capabilities. Found that a person's level of creativity didn't play a role in how they perceived the software's utility. Perceived usefulness was not significantly more influential than ease of use (suggesting complexity is a major barrier).	Integrated UTAUT	Professionals and participants from the public and private sectors, with a specific highlight on IT professionals (experts).	A comprehensive integrated model that combines: 1- UTAUT (Unified Theory of Acceptance and Use of Technology) 2- OSS Attributes (Quality, Compatibility, Security) 3-Infrastructure factors and user trust.	Alrawashdeh et al. (2020)	[8]

Key Findings	Methodology	Sample Characteristics	Theoretical Models	Author(s) & Year	Ref
Intrinsic motivations (Autonomy and Social Connection /Relatedness) significantly improved how students perceived the software's usefulness and ease of use. Perceived usefulness and ease of use acted as mediators that translated psychological motivations into a concrete willingness (intention) to use the software. Mandatory training was found to be a successful tool for strengthening these intrinsic motivations, making OSS a more viable alternative to commercial software.	Quantitative / Empirical.	Students who participated in a mandatory training program on open-source software.	A hybrid conceptual model combining: Self-Determination Theory (SDT): Specifically, the elements of Autonomy, Competence, and Relatedness (Social Relationships). Technology Acceptance Model (TAM): Perceived Ease of Use (PEOU), Perceived Usefulness (PU), and Behavioral Intention.	Racero et al.(2020)	[9]
Access to training has a direct, significant positive impact on both perceived ease of use (PEOU) and perceived usefulness (PU). The visual presence and exposure of OSS in the learning environment significantly boost how "easy to use" students perceive the software to be.	The study conducted a pilot test followed by a questionnaire administered to university students.	University students in a developing country.	extended Technology Acceptance Model (TAM), expanded it by adding external variables: social, personal, and environmental factors, with a specific focus on training and visibility.	Ebardo (2018)	[5]

Key Findings	Methodology	Sample Characteristics	Theoretical Models	Author(s) & Year	Ref
The study identifies low cost of ownership, source code access (credibility), and regulatory compliance as the primary reasons for successful adoption in the Omani context. FOSS significantly reduced software expenditure and improved the flexibility of content exchange. This case study treats factors like security and ease of use as inherent advantages that motivated the transition, rather than variables tested for "significant effect" in the statistical sense.	Qualitative case study.	Higher education context (The College of Applied Sciences in the Sultanate of Oman), specifically the IT major's courses and curriculum.	Educational framework specifically designed for curriculum reform. It focuses on administrative and organizational pillars: technical planning, committee formation, training, policy development, and awareness.	Al-Hajri et.al (2017)	[6]

included in the review. Studies lacking sufficient analytical rigor to maintain focus and control were rejected.

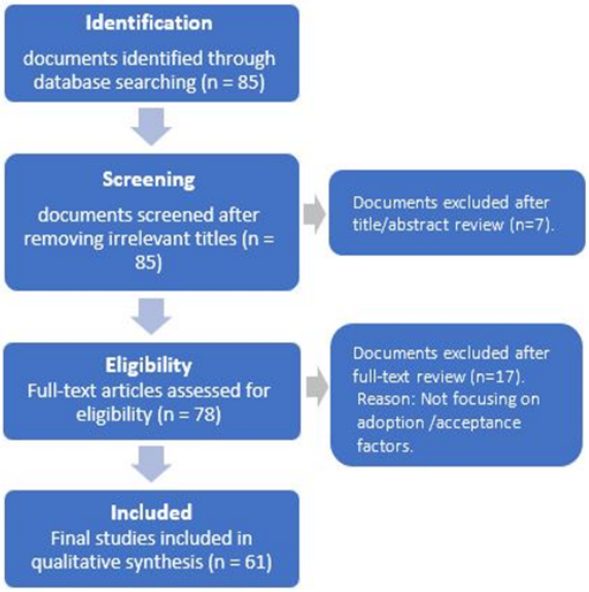

Figure 1. Flow chart of the selection process.

3.4. Data Synthesis and Conceptual Framework

After a thorough review of the 61 included studies, the findings were systematically organized into six conceptual categories. This structured approach facilitated a comparative examination of open-source application adoption across diverse fields such as academic disciplines, public administration, and enterprise resource planning (ERP) environments. By synthesizing these comprehensive perspectives across various fields, a unified conceptual model was created to map the multifaceted environment influencing free software adoption.

3.5. Quantitative Analysis

Owing to the qualitative nature of this assessment, which focuses on identifying the essential factors for the adoption and acceptance of open-source software, no aggregate statistical analysis was performed.

4. THEORETICAL FOUNDATIONS AND KEY FACTORS INFLUENCING OSS ACCEPTANCE

The Open-Source Initiative (OSI) defines open source as access to source code that has the following qualities: free distribution, no discrimination against individuals or groups, no discrimination against fields of endeavor, a license that is not product-specific, a license that does not restrict other software, and a license that is technology neutral [12].

Building on this foundation, understanding OSS acceptance requires examining behavioral and motivational theories that explain how users form the intention to adopt technology. Among the most widely used frameworks in this domain is the Technology Acceptance Model (TAM), which posits that Perceived Usefulness (PU) and Perceived Ease of Use (PEOU) are the primary determinants of Behavioral Intention (BI) to use a system [8], [13].

PU reflects the extent to which users believe that OSS will enhance their performance, whereas PEOU assesses the effort expected to learn and use the software. Numerous OSS studies confirm that both constructs positively affect users' intentions to adopt, especially when combined with contextual elements such as training, visibility, and technical quality [5], [14].

Beyond these behavioral dimensions, the technical characteristics of OSS also play an essential role in shaping user perceptions. In addition to TAM, System Quality has emerged as a critical dimension of OSS acceptance. It encompasses factors such as security, reliability, interoperability, and performance [1]. High system quality not only improves user satisfaction but also builds trust in OSS solutions, particularly in mission-critical environments. Furthermore, recent research emphasizes the role of psychological and cultural factors in shaping OSS acceptance. Social identity, motivation, and familiarity with OSS or proprietary systems influence users evaluation of software alternatives [13]. These insights underline the necessity of incorporating both technical and human-centered dimensions when evaluating OSS's acceptance.

Supporting these theoretical insights, empirical evidence highlights the significance of these factors. The results from the structural equation modeling show that there is a statistically significant positive link between the acceptance of open-source software and several factors, including how easy users find the software to use, how useful they think it is, their intention to use it, the quality of the output, how well it works with other software, its functionality, and how easy it is to maintain it. Unlike most previous studies that did not find a positive connection between visibility, trialability, and the acceptance of open-source software, this study found that there is a positive relationship between these factors and acceptance [15].

At the organizational level, further studies have highlighted broader institutional patterns influencing OSS adoption. The main research findings were as follows: (a) the Carnegie Classification plays an important role in understanding the level of awareness, use, and views of open-source software; (b) CIOs were much more likely than CAOs to know detailed information about open-source software and to be at different stages of using various open-source applications; (c) IT administrators' primary motivations for adopting open source software are focused on reducing ownership costs, while account-

ing administrators focus primarily on supporting active learning; and (d) no significant progress has been made in formulating formal policies and procedures to ensure that open source software applications comply with relevant federal and state regulations, nor have formal policies and procedures been developed to ensure the security of these applications [16].

The literature indicates that the acceptance of Open-Source Software (OSS) is influenced by a complex interplay of technical, psychological, and sociocultural factors. Based on the synthesized findings of the reviewed studies, six core categories emerged as the most influential: perceived usefulness and ease of use, social influence and organizational support, training and visibility, technical quality and usability, cultural and motivational, and technological and economic factors.

Key Factors Influencing OSS Acceptance

4.1. Perceived Usefulness and Ease of Use

The results from the structural equation modeling show a strong positive connection between how people accept open-source software and several factors. These factors include perceived ease of use, perceived usefulness, intention to use, output quality, compatibility with other tools, features, and ease of maintenance. For users to accept open-source software, they must view it as an effective tool for accomplishing their daily tasks and one that is easy to use.The study showed that perceived ease of use (PEOU) significantly and positively affects perceived usefulness (PU) ($\beta = 0.208$, $p < 0.01$), and that ease of use has a positive effect on the intention to use open-source software ($\beta = 0.181$, $p < 0.001$). Moreover, intention to use has a positive and significant effect on actual usage behavior ($\beta = 0.304$, $p < 0.01$). The results confirmed that users in Thailand perceive open- source software as easier to use and potentially more useful than proprietary software [15].

Moreover, another study [13] consistent with the Technology Acceptance Model (TAM) found that perceived usefulness (PU) and perceived ease of use (PEOU) significantly influenced the intention to adopt OSS applications. Interestingly, PU was relatively weaker than PEOU in explaining adoption intention. Moreover, personal innovativeness in information technology (PIIT) showed a positive relationship with both OSS adoption intention and PEOU, highlighting the importance of individual innovativeness in promoting OSS adoption. In particular, perceived ease of use (PEOU1) was described as the ease with which most tasks are accomplished when using open-source software, compared to proprietary software. Similarly, perceived usefulness (PU1) was a significant factor in adoption decision-making. Additionally, personal technological innovation (PIIT1) was identified as a key factor influencing open-source software adoption, reinforcing the role of individual characteristics in its acceptance.

However, personal constraints, such as self-initiated innovation in IT, social dependency, perceived ease of use, and perceived usefulness, may emerge as barriers in some cases [17].

Furthermore, performance expectancy (PE) is considered a measure of users' belief in the feasibility of using a software system for their tasks, whereas effort expectancy (EE) reflects the ease of use and convenience of the system. Research has shown that effort expectancy positively affects performance expectancy, which in turn positively influences behavioral intention (BI) toward using open-source software. Commercial research has also shown that BI has a significant positive effect on actual use, indicating that open-source software requires less effort and is more beneficial than commercial alternatives [8].

At the organizational level, directly or indirectly adopting OSS as part of a firm's business strategy may allow the firm to accelerate its internal innovation process. This can be attributed, in part, to the availability of external labor, which may result in lower internal maintenance costs, better product quality, and quicker time-to-market. The third possible benefit is the infusion of features from the OSS ecosystem [18].

4.2. Social Influence and Organizational Support

Societal factors and entities play a key role in determining the diffusion of open-source software, particularly in educational and organizational settings. Experiments have shown that low costs and positive social impacts, such as peer recommendations or institutional policies, significantly impact guiding behavior toward using this software in government departments [1].

Similarly, low initial installation and implementation costs, infrastructure availability, and the opinions of others (trainers, colleagues, and superiors) are crucial factors in driving OSS adoption among IT professionals [8]. Furthermore, social influence (SI) represents how individuals perceive the opinions of others, especially friends and colleagues. This study also demonstrated a close relationship between social influence (SI), self-confidence (SE), supportive conditions (FC), operating costs (COS), and behavioral intention (BI) [8].

In addition, Gwebu and Wang [13] demonstrated that social identity and active participation in OSS communities enhance user confidence and strengthen acceptance outcomes. However, although social influence is commonly considered a key determinant of technology acceptance, one study suggests a more nuanced view of OSS: Users do not start using this technology because they are influenced or pressured by the people or situations around them. Instead, their reasons for using it

are based on other factors, such as wanting to be recognized within the open-source software community. This recognition is separate from what others think of them [19]. Another study found that there is no strong link between what others think of them, how useful they believe the software is, or how they see themselves in social terms. This shows that, overall, users choose to use open-source software for their own reasons, not because of what their friends or peers think, and they do not feel like it makes them look better in front of others [15].

Moreover, any user can choose to become a co-developer, and the success of an OSS project is directly correlated with the level of community building [11]. The total existing user base of previous editions of an OSS project is positively associated with the market success of its current version [20].

Nevertheless, no positive relationship was found between job relevance and PU or between result demonstrability and PU. A possible explanation is that users do not believe that using OSS reinforces or enhances their job-related tasks and that tangible results are not evident to them. It may be that their workplace or organization relies heavily on proprietary software that is installed and commonly used, which may prevent them from having the opportunity to use open-source software to perform their work tasks [15]. Likewise, in terms of software characteristics, no positive relationship was found between software reliability and PU. It could be that Thai users do not focus on OSS being reliable or stable when compared to its proprietary software counterparts [15].

Research has also shown that the organizational adoption of OSS has demonstrated that its use was frequently a bottom-up initiative, promoted by technical staff within the organization who are adherents to the open-source movement. In certain circumstances, decision-makers may be viewed as supporters of the open-source movement. These workers will operate as boundary spanners for their companies, introducing innovative technology [21].

Additionally, an article shows that open-source adoption depends crucially on organizational IT capabilities, network effects, and the fit of OSS with an organization's application needs [22]. As reported in the study [23], the factors of the selection, evaluation, and adoption processes were classified into eight main groups, including Community and Adoption, Support, and Service. Similarly, another study focused on the adoption of OSS within Kazakhstan's public organizations, using the Ministry of Oil and Gas as an example. Questionnaires and interviews were used to understand the current state of open-source software (OSS) in Kazakhstan and assess the readiness of organizations to adopt it. OSS policies must include four main pillars: technical, organizational, administrative, and individual. These pillars are essential for building effective strategies, as they represent the key pillars of the adoption process. Strategic activities should be formulated accordingly. Adoption is a sequence of steps that should be performed; it requires a specific duration of time [24].

Furthermore, the results of this study [25] showed that performance expectancy, effort expectancy, social influence, and supportive conditions significantly impact the intention to use an ERP system. However, this study did not prove a direct, statistically significant relationship between supportive conditions and actual system use. This study explores the potential impact of education and organizational size. The results indicate that ERP is used in all types of organizations, regardless of their size and the level of education of their employees. However, the results confirm that there is a direct positive effect of the size of the organization on the extent of actual ERP use.

Finally, these mixed findings highlight the contextual nature of OSS adoption, suggesting that social and organizational factors may vary in influence depending on cultural setting, institutional readiness, and user motivation.

4.3. Training and Visibility

The adoption of OSS is fundamentally dependent on awareness. Survey findings revealed that 84.4% of respondents were aware of the existence of OSS, while 98% acknowledged its free availability, and only 2% remained oblivious to this trait. These statistics highlight the importance of informative exposure and visibility in promoting user acceptability and OSS adoption [26]. In addition, this study highlights various aspects that influence the adoption of OSS in the IT community, including cost-effectiveness, availability of source code, community support, security, and ease of use. The findings show that the decision to embrace OSS is heavily influenced by awareness and perceived benefits [26].

Several studies have emphasized the importance of structured training and visibility in enhancing user perceptions of OSS. Ebardo [5] showed that students who received formal training exhibited more positive attitudes toward OSS usability than those without such exposure. Racero et al. [9] found that frequent exposure to OSS in academic environments improved student perceptions and facilitated the transition from proprietary to open-source platforms. These findings underscore the value of integrating OSS into educational curricula and organizational systems as part of broader adoption strategies.

Moreover, training and environmental exposure to OSS are among the most frequently cited factors influencing users' readiness to adopt new systems. This study [5] illustrates their impact in the educational context. It is based on the idea that both the learning or training experience and the prevalence of open-source software (OSS) in the environment influence students' decisions to use this software. This study examines the visibility of open-source software and the extent of

training students receive, and how much they adopt it, from the perspective of university students in the Philippines. The Technology Acceptance Model (TAM) was used, but it was modified to include additional factors. Its accuracy was checked using a method called partial least squares within structural equation modeling. The results show that the TAM model performed well in this situation. Training was found to have a positive effect on how easy students thought using the software was and how useful they found it. Training also helps improve the perceived ease of use, which supports students in accepting open-source software.

Furthermore, the study [14] highlights how external determinants, particularly user training and support, influence behavioral intentions. The primary objective of this study was to investigate the impact of open-source software (OSS) training on end users' acceptance and willingness to use OSS at various educational stages. This study examined how OSS training affects students' acceptance and use of OSS solutions. It was found that OSS training helps students use these solutions more. The study also provides suggestions for both people who make OSS and those who use it. One key point is that students who receive training are more likely to accept and use OSS solutions than those who do not. This is because they find the system useful and easy to use in their research. Another important finding is that training, how well the tool fits the user's needs, how complex the technology is, and how much support the trainer provides are all important for successful adoption. This study had two main limitations. First, only Spanish students were included; therefore, including other groups would make the results more comprehensive. Second, the link between training and how useful the system is not very strong, so the idea might require more work.

Additionally, users with good knowledge of MS Office did not have any trouble converting to OpenOffice. Most of the issues were caused by employees who knew very little about the office [27]. Lack of competence was named among the top six technology risks by IT decision-makers [28].

Finally, unlike most studies that did not find a link between visibility and trialability when it comes to accepting open-source software, it is interesting to note that the study [14] did find a positive connection. Visibility had a positive effect on PEOU (β = 0.135, p < 0.001), and trialability had a positive effect on PEOU (β = 0.155, p < 0.001). The most likely reason for this is that users are now more familiar with open-source software and are more willing to try it [15].

4.4. Technical Quality and Usability

System quality and usability remain fundamental to OSS acceptance, particularly in enterprise settings. These dimensions include security, interoperability, performance, and customizability [1]. With a significant expansion in the non-technical users of OSS, expectations connected to greater software quality will develop [29] .

In support of these observations, several OSS-specific quality indicators have been evaluated. [8]

· **Cost-effectiveness (COS)** refers to comparing the benefits people see from using software with the cost of using it, including the cost of installing the software and obtaining support services.

· **Security (SEC)** refers to the number of people and organizations that trust the software to be secure.

· **Software quality (SQ)**. This is achieved through the quality of services, which mainly involves how fast they work, how reliable they are, and how well they protect personal information.

Furthermore, researchers have stated that a major advantage of open-source software is its reliability. In software, reliability means that there are fewer mistakes or issues that can lead to things not working properly, data being lost, or the system crashing. Open-source software is trustworthy because programmers can regularly update and improve it because they have access to the source code [30]. Numerous authors have noted that OSS public inspection aids in the early detection of software flaws and security issues, thereby lessening their impact [27].

Additionally, customizability can affect the reliability. Some argue that because the program's source code is open and available to everyone, individuals or groups can change it to fit their requirements. This is especially important for companies in certain industries, such as healthcare and government agencies. Compared with customizing closed-source systems, modifying open-source software is much easier. The customized version can then be shared with the open-source community, where it can be tested, improved, and later returned to those organizations [30]. The open-source design of the suggested framework makes it easier to modify and adapt it to specific remote sensing applications or other domains. It provides customers with access to both the underlying software code and manufactured hardware, allowing them to change, adapt, and extend the platform's capabilities to meet their individual needs [31].

One of the main quality factors influencing user consent and OSS sustainability is usability, which has been identified as the key to OSS success in the literature. Because unusable software is not viable, usability issues must be considered. Many authors have admitted that OSS usability is inadequate because developers typically write their own software, resulting in subpar software usability. Such poor usability restricts the adoption of OSS and undermines its sustainability. Unfortunately, no agreement has been reached regarding its definition; leaving out important criteria and keeping unimportant ones will undoubtedly cause the usability evaluation to go in the wrong direction [11]. Understandability, learnability,

operability, and attractiveness are other subcategories of usability [29]. Dawood et al. [10] Notably, many OSS platforms suffer from limited documentation and a lack of systematic usability evaluation processes, which undermines their adoption potential.

Moreover, Raza et al. [32] observed that poor user experiences—stemming from interface complexity and limited access to support—discouraged usage despite the presence of strong technical functionality. However, their study also demonstrated that implementing features such as help features and user training significantly improved satisfaction and acceptance. These factors help users overcome difficulties and contribute to software improvement, thereby enhancing the overall experience.

Similarly, in a related study [33], the hypothesis testing results indicate that interactive assistance features and the ability to report usability bugs are positively and significantly correlated with OSS usability. These factors help users overcome difficulties and contribute to software improvement, thereby enhancing the overall experience.

Based on these findings, learner characteristics have been integrated into adaptive e-learning systems. to provide customized support and recommend suitable training content [34].

In addition, software interoperability, quality, and security all have positive and significant impacts on performance expectancy. This means that users believe that open-source software (OSS) works well with other software and can improve its overall performance. Users also think that good quality and strong security in OSS can make it more useful, reduce problems when using it, and help solve security worries [8]. To deepen the understanding of user-centered factors in OSS acceptance, one study [19] extended the traditional TAM by including external constructs. Software quality, system capability, and flexibility play important roles in how users perceive the usefulness and ease of use of open-source software. This study identified four factors outside the traditional TAM model: software quality, system capability, social influence, and software flexibility. The research shows that OSS developers should consider how future users might accept their software. According to the study, users want OSS that is high quality, has a wide range of features, and is flexible, because they believe these qualities make the software easier to use and more valuable. The findings also provide two main practical lessons for users and organizations. First, to encourage more OSS use, users and organizations should choose software that is both useful and easy to use. Second, when selecting the best open-source software, they should take into account factors like software quality, system efficiency, and flexibility. Overall, these features help to increase user acceptance of open-source solutions.

Furthermore, the study [15] also found that output quality has a positive effect on perceived usefulness (β = 0.073, p < 0.05). This means users believe open-source software (OSS) is of high quality and is more than enough for their daily work. Compatibility was also found to positively influence perceived usefulness (β = 0.148, p < 0.001). This suggests users find OSS works well with their jobs and fits into their current situation. They also feel it does not take much effort to learn or change their habits to use it. The study also found that software functionality and perceived ease of use have a positive relationship (β = 0.078, p < 0.05). This means Thai users think the overall features of OSS are easy to use, which is different from the common belief that OSS is hard to use. The study also found a positive relationship between software maintainability and perceived usefulness (β = 0.089, p < 0.05). This shows that users in Thailand see OSS as highly beneficial because it is easier to modify and fix bugs. OSS is less expensive to maintain compared to proprietary software because it benefits from the contributions, knowledge, and skills of open-source communities. Sharing and exchanging knowledge is valuable for users because they do not need to pay large amounts to proprietary software companies for maintenance, unlike proprietary software, which uses about 70% of software budgets for maintenance.

Other studies have classified OSS adoption factors based on practical software characteristics. The factors reported in the study [23] were classified into eight main groups: development process, functionality, operational software characteristics, and quality. Large-scale studies have also found that how well a system works and how easy it is to use are some of the top reasons people choose open-source software. The results show that using free and open-source software is linked to low cost, good performance, influence from others, and good system quality. This link is mainly because of how easy the software is to use, how secure it is, and how well it works with other systems, which are important factors when people decide to adopt open-source software [30].

Finally, the results validate the assumptions that the specified dimensions of the study model (system quality, security, interoperability, usability, costs, effort expectancy, performance expectancy, and social influence) have a strong influence on the intention to use FOSS. System quality is a vital component integrated into the model, and its strong influence is due to factors such as security, consistency, and ease of use. Additionally, the study showed that the low cost of FOSS, expectations of performance, and social influence all contribute significantly to its diffusion in the public sector. In contrast, the study participants did not consider effort-related estimates to be a significant factor in FOSS acceptance [1].

4.5. Cultural, Psychological, and Motivational Factors

Cultural perceptions and intrinsic motivation play important roles in shaping users' attitudes toward open-source software, especially in developing countries. Building on this idea, understanding individuals' willingness to use open-source software and computer self-efficacy (SE) is a key psychological factor. SE refers to users' confidence in accomplishing a specific task using a specific computer program. Similarly, facilitating conditions (FC), defined as the extent to which individuals perceive that available resources and support are required to use software systems, play an important supporting role [8].

Furthermore, Bhatiasevi et al. [15] revealed that users in certain cultural contexts perceive free software as lower quality, which creates psychological resistance to adoption.In alignment with behavioral models, intention to use has been conceptualized as a psychological measure of readiness. Racero et al. [9] showed that students who felt in control of their learning and were capable of navigating OSS tools were more inclined to adopt them. These findings reinforce the importance of fostering positive, intrinsically motivated perceptions to expand OSS acceptance. Self-determined motivational factors, such as autonomy, competence, and relatedness, play a critical role in OSS acceptance.

According to the basic ideas behind the acceptance of software systems, behavioral intention refers to how willing users are to perform a certain action [8]. In this regard, researchers have discussed the reasons people participate in open-source initiatives. They claim that internal factors and external rewards are the two main categories into which people's motives can be divided. social incentives, communal motivations, and reward motivations [35].

Expanding on user autonomy and freedom, OSS empowers users with greater choice and control over their software. [30]

· **More end-user choice and control:**

Open-source software provides an alternative to proprietary software owned by a company. It offers users more choices and continues to improve over time. These software packages can compete with many other products sold by companies.

This helps users avoid depending too much on one company that controls the source code and decides which hardware to use [30].

Closely related to cultural and motivational appeal is the low-cost advantage of OSS, which often resonates with institutional values and individual preferences in cost-sensitive environments.

Little cost: The most important aspect of open-source software (OSS) is that it can be used for free or at a low cost, which is why many popular software products are open-source. According to Titterton, 70 percent of business decision-makers are interested in using OSS because it helps save money. The fact that open-source software is available for free or at a low cost is thought to be a big reason why it is adopted so quickly [30].

Finally, licensing frameworks also contribute to shaping the motivational and legal perceptions surrounding OSS.

Licensing: In 1997, when the "Open-Source Software Definition" was created, certain rules were set to explain what makes open-source software different from other software types. These rules include laws that control how software can be licensed for use. There are four main parts that help define these licenses: first, whether they allow mixing open-source software with proprietary (closed-source) software; second, whether changes can be made privately but not shared back with the original creator; third, whether anyone can share the software with others; and fourth, whether the original copyright owner has any special rights. Because of these advantages, large companies, governments, and schools that cannot afford expensive licenses often choose cheaper open-source options [30].

Moreover, collectiveness, cooperation, and transparency are concepts and objectives that OSS shares with eGov in its most advanced stage of development. These common principles imply that OSS expansion may establish the proper intellectual framework and social conditions for e-government maturity [36].

In addition, Zeithaml distinguishes between intrinsic and extrinsic cues in decision making. Intrinsic cues are product-related features that cannot be altered without modifying the product itself, whereas extrinsic cues serve as surrogates to reduce perceived risk. Cue Utilization Theory (CUT) explains that OSS consumers and developers respond differently to these cues, influencing OSS adoption and project success [20]. Studies also indicate that native language interfaces significantly improve software usability, as users perform better when the software is presented in their mother tongue [20].

Another important factor in the development of free software was, and still remains, its ideology. The right of individuals to have access to programs that they need and the ability to use and manage the code as they see fit is a strong opinion among many free software developers [37]. The impact of the open-source model on the human element may also be substantial when compared with the use of proprietary "closed" software, including proprietary Unix. From a 'passive' licensee (with no real power on the tool's quality), the development team will feel promoted to a 'member of a community of peers', contributing to forums and exchanging codes and tips [38].

4.6. Technological, Organizational, and Economic Factors

Within the broad context of OSS adoption, several studies have examined the roles of technological, organizational, and economic elements. Technical, business, and economic factors play a role in the use of open-source software. However, factors that induce trust, such as reliability and degree of satisfaction with functional requirements, may differ for these two types of products. This study analyzes the extent to which the factors that determine open-source software acceptance differ from those that induce trust therein, on the thought that while support is also a major factor for open-source software adoption, the trust factor has a technical factor that takes precedence over the rest. In this study, we categorized relevant factors of the technology acceptance model and trust and surveyed respondents to determine whether the degrees of importance assigned to them differed [1].

Expanding this perspective, a comprehensive study identified twenty-two adoption factors, categorized into three main groups: technological (nine), organizational (nine), and economic (four). These factors provide a solid foundation for understanding the diverse influences on OSS adoption [39]. The factors listed below are:

· **Technological Factors:**

Many technological factors are listed, such as usability, portability, compatibility, reusability, maintainability, documentation, trialability, customization, and reliability.

· **Organizational Factors:**

Support, aid from upper management, vendor lock-in, training, a change-oriented attitude, successful FLOSS Adoption Cases, time to adopt, IT, and business process reengineering are all crucial organizational factors.

· **Economic Factors:**

Many economic factors are listed, such as licensing costs, total cost of ownership (TCO), operational costs, and support costs.

Aligned with these categories, empirical studies have examined adoption in specific organizational contexts. A previous study by Yaseen and Bahari in 2014 stated that factors such as organization size, cost, human resources, management support, user resistance, and the benefits of technology play a role in deciding whether to adopt open source software. Gurusamy and Campbell (2011) pointed out that when businesses consider using open-source software, they focus on aspects such as compatibility, economic value, quality of documentation, organizational environment, legal concerns, product quality, user impact, and data conversion processes. Research shows that when organizations decide to use open-source software, it is heavily influenced by how well the software works with their current technology and the skills of their employees. The study found that having support available and avoiding compatibility problems with open-source software can prevent organizations from adopting it. Gichira, Miriti, and Kahonge (2012) and Johnston, Begg, and Tanner (2013) examined how different factors, such as technology, organization structure, and the environment, can either help or hinder the use of open-source software in companies [40].

Additionally, four factors influence the adoption of open-source software. These include managerial involvement, social influences, and organizational support. In contrast, uncertainty avoidance is the only national-level factor that influences open-source software adoption [40]. In addition, the factors mentioned in the study [19] were classified into eight main groups, including economic and licensing factors.

At the enterprise level, one study explored why companies choose open-source ERP solutions. The adoption frequencies were as follows: Be Competitive 35%, Flexibility / Avoid Private Software Limitations 15%, Software Continuity 11%, Financial Criteria 12%, The company did not use an Open Source ERP solution 12%, Other purposes 15% [2].

Furthermore, researchers have aimed to understand why these companies choose open-source solutions. The main reasons were gaining competitiveness (34.6%) and the flexibility of the software. Initially, the researchers thought that cost (Financial Criteria) was the main reason. However, companies saw the system as part of their strategy, and the main reason for using the software was still competitiveness (34.6%). Researchers have also attempted to determine which features of the system influenced the decision to choose it. The most liked features were a user-friendly interface, an active community, easy customization, and scalability. Polarization is not limited to just a few features that satisfy the sample. Friendly Interface 19%, Active Community 15%, Scalability 12%, Easy Customization 12%, Documentation Availability 11%, Other 31% [2].

In the context of developing countries, open-source software can be useful for developing countries in two ways: as a means of reducing licensing costs and boosting local technical development by possessing the source code of these goods [35]. Case studies in the public sector likewise show that OSS can lower the cost of ownership and reduce vendor lock-in by purchasing open-source software (OSS) rather than proprietary software. Overall, the benefits of Open Source substantially outweigh the associated costs. These advantages are primarily related to labor cost savings and openness (including independence and standards), not to increased income generation [41].

Consistent with this, the main reasons given are independence from software suppliers, compatibility and conformance with open standards, cost-effectiveness, transparency and ability for customization, security and dependability, interoperability, and availability of OSS community assistance [36].

When evaluating OSS ERP systems, Johansson and

Sudzina identified several factors for organizations to consider when selecting the best OSS ERP to meet their needs, including speed and ease of implementation, product price, vendor support, dependability, ease of use, customization capability, integration capability, fit to organization, flexibility, training offered, use of the newest technologies, upgrades, and scalability [12].

Regarding OSS success, studies have found that projects that utilize a non-restrictive license have more commercial success than those that use a restrictive license, and technical success is better in OSS initiatives that assign responsibility [20]. The data also show that a company's proportionate involvement in OSS increases with its size (businesses with 50 or fewer people made nearly half of the commits in our sample of the most active organizations in OSS). While the Information and Communication Technology (ICT) sector accounts for over half of all contributions (8% of all employees participated in OSS development European Union (EU)-wide), professional, scientific, and technical companies as well as, to a lesser level, wholesale, retail, and financial companies were also heavily involved [41].

From an individual and community perspective, motivations for participating in OSS include finding technological solutions, avoiding vendor lock-in, advancing the state of technology, producing high-quality code, information seeking, knowledge production, and individual participants' personal interests were the top reasons to engage in OSS [41].

In terms of risk, large companies do not consider hidden costs as a major risk factor, whereas medium-sized enterprises view them as more significant and requiring careful evaluation [28]. According to Riehle, costs are one of the reasons businesses use open source. However, he states that the open-source cost perspective is primarily a justification for solution providers. Solution providers gain the most from open-source software because they enhance earnings through direct cost reductions and the potential to reach more consumers through better pricing flexibility [35]. In the field of enterprise resource planning, the use of open-source software (OSS) has grown recently for a number of reasons, including cost savings and ongoing maintenance, support, and enhancement [12].

Within ERP environments, the literature indicates that Small to Medium Enterprises (SMEs) benefit from cloud-based ERPs more readily because of the low capital expenditure and quicker time to market. However, many of the problems and difficulties revolve around data security, confidentiality, and worries about moving mission-critical apps to the cloud, which are frequently not SMEs' main [42].

At the macroeconomic level, enterprises in the EU invested approximately €1 billion in OSS in 2018. The analysis indicates that the OSS pool contributes significantly to the European Union's gross domestic product (GDP), and that an increase of 10% in contributions would create between 0.4% and 0.6% additional EU GDP per year. Additionally, the study finds that a 10% increase would result in over 600 new ICT start-ups in the EU annually [41].

To synthesize these findings, Vargas et al. provided an explanation for the classification of a certain factor. Technical considerations are elements linked to the release process, code quality, and functionality. The community surrounding the project, individual perceptions, and other facets of the organization where the package is generated are considered organizational influences. Economic considerations involve the financial aspects of package selection, such as licenses, total cost of ownership, and risks. Technical variables are frequently observed in the literature. These variables were present in 49 of the 54 studies and were related to the technical features of software programs. The most prevalent and significant impact elements are software compatibility, dependability, usability, and customization [43].

Building on these classifications, the second category of elements relates to organizational aspects of the project. Although slightly less represented than technical variables, they remain highly visible in the literature. Notably, "support" —the most influential factor—belongs to this category and was reported 45 times across the 54 reviewed studies. The last category addresses economic factors. Although these appear less frequently in the literature, they remain important considerations for software practitioners [43].

Three main conclusions were drawn by the researchers. When choosing a package, software engineers first distinguish between an adoption factor and a trust factor. Second, the software engineers in this study come to the conclusion that organizational elements are more significant than technical considerations, which are typically regarded as the primary trust factors in literature. Ultimately, they discover that it is impossible to develop a single, cohesive perception of trust since different types of software engineers require diverse perspectives on trust. [44]

Complementing these perspectives, developer download behavior can be understood through two contextual scenarios: In the first context, which relates to control by license type — a cultural legacy of open-source sellers to enterprises — developers and end users find licenses more appealing when they are less restricted. In parallel, the second context is the control by the operating system environment, where modules are cultural heritage. For example, operating system components, such as libraries, compilers, build tools, debuggers, and version controls, developed in the C programming language, have a high possibility of recruiting more developers [45].

5. BEHAVIORAL MODELS APPLIED TO OSS ACCEPTANCE

The use of behavioral intention models has become essential for analyzing users' decisions to adopt open-source software, focusing on the motivations and mechanisms underlying these choices. These models provide structured frameworks for assessing user behavior and predicting their intentions. This is evident in fields such as education, government, and other organizations. Prominent examples include the Technology Acceptance Model (TAM), Unified Theory of Acceptance and Use of Technology (UTAUT), and Self-Determination Theory (SDT).

5.1. Technology Acceptance Model (TAM)

The Technology Acceptance Model (TAM) is relevant for evaluating open-source software selection. This model is based on two main components: perceived usefulness (PU) and perceived ease of use (PEOU), both of which influence behavioral intention (BI), which in turn leads to actual use [5], [8], [9].

Racero et al. [9] combined the technology acceptance model with self-determination theory to explore students' behavioral intentions after receiving open-source software training. Their research results indicated that both the acceptance model (PU) and the perceived ease of use (PEOU) model had a significant impact on students' propensity to adopt open-source software, especially when the training enhanced their feelings of familiarity and trust. Ebardo [5] further confirmed this by showing how structured exposure and visibility improved users' perceptions of ease of use and perceived usefulness.

Alrawashdeh et al. [8] extended the scope of the TAM model by adding variables relevant to open-source software, including software security, quality, and interoperability.

5.2. Unified Theory of Acceptance and Use of Technology (UTAUT) and Hybrid Models

The Unified Theory of Acceptance and Use of Technology (UTAUT) model builds on its predecessor, the TAM model, with the addition of important factors. These factors include performance expectancy (PE), effort expectancy (EE), social influence (SI), and facilitating conditions (FC). Together, these elements influence individuals' intentions and actual use of technology.

Regarding open-source software, Alrawashdeh et al. [8] developed a comprehensive model. This model integrates aspects of the UTAUT with open-source software attributes. This model is effective in predicting user performance outcomes. The results show that software quality, security, and interoperability significantly influence the performance requirements. Furthermore, cost and facilitating conditions emerged as critical factors in the adoption process, particularly in resource-constrained environments.

Gallego et al. [14] Support the importance of external support and ease of use in OSS acceptance. This aligns with the key concepts found in behavioral models such as TAM and UTAUT.

According to a study [25] applying the UTAUT model, "performance expectations, effort expectations, social influence, and facilitating conditions are important in the context of enterprise resource planning (ERP), and they help increase users' interest and willingness to use ERP. Their use can also influence these factors. The study found that 'encouraging conditions' have little impact on actual ERP implementation and that the relationship between these factors is not very strong. These results show the importance of the main parts of the UTAUT model in influencing user behavior, highlighting the different and sometimes conflicting influences of environmental conditions on actual performance.

While UTAUT emphasizes external and social factors, SDT provides insights into intrinsic motivational influences on OSS adoption.

5.3. Self-Determination Theory (SDT)

Self-determination theory (SDT) focuses on the role of intrinsic motivation, specifically autonomy, ability, and belonging, in determining the behavioral outcomes. This theory complements older models by considering the psychological needs that influence user behavior and satisfaction.

This study [9] examines the use of open-source software (OSS) in education. A theoretical model based on Self-Determination Theory (SDT) and the Technology Acceptance Model (TAM) was constructed to investigate and identify open-source software usage. This model consists of six main components: (1) sense of autonomy, (2) sense of efficacy, (3) sense of belonging, (4) perceived ease of use, (5) perceived desirability, and (6) behavioral intention to use the device. Intrinsic motivation, that is, autonomy and belonging, positively influences the perception of the usability of open-source software, which is positively reflected in the behavioral intentions towards using this software.

In a joint TAM and SDT study, Racero et al. [9] demonstrated that intrinsic motivation is a key factor in students' perceptions of the suitability and ease of use of open-source software, which enhances their enthusiasm for learning. Students who felt in control of their learning and enjoyed freedom of action in the classroom environment were more likely to encounter difficulties using the software. These findings underscore the importance of careful planning for the implementation of open-source

software, which should not be limited to technical and administrative aspects but should also focus on supporting users' feelings of competence, autonomy, and social engagement.

6. CHALLENGES AND BARRIERS TO OSS ACCEPTANCE

Despite the growing interest in open-source software and its use in various fields, some obstacles and pitfalls hinder its widespread and fruitful application. These challenges include technical, administrative, and political aspects that affect users, whether they are individuals or institutions.

6.1. OSS Usability Challenges from Stakeholders' Perspectives

In contrast to the standard OSS strategy, quality assurance must be completed before the program is provided because users are not co-developers capable of finding and fixing flaws [29].

6.1.1. Technical Complexity and Documentation Issues

The OSS development process has unique features (e.g., community members are geographically dispersed, resources are scarce, and interaction designers may not be familiar with the culture). This hinders the straightforward adoption of several HCI usability strategies [46].

In addition, technical complexity remains a major boundary for the selection of open-source software. Many OSS solutions suffer from a lack of documentation, making installation, configuration, and troubleshooting difficult [1], [10], [33]. The lack of comprehensive and easy-to-use manuals often discourages non-expert users and organizations from fully engaging with OSS products. Furthermore, the lack of technical support and community responsiveness can increase user frustration, leading to a loss of trust and eventual abandonment.

6.1.2. Usability Barriers and User Interaction Limitations

Usability testing is an essential component of the software life cycle. Holzinger argues that "the earlier critical design flaws are detected, the more likely they can be corrected" and stresses the importance of usability testing early in the software life cycle [47].

Consequently, the design priorities of OSS projects are directly affected by the low participation of usability experts. Despite the growing popularity of user-centered designs in OSS, usability is still not considered one of the main goals in many design scenarios [29].

In addition, a straightforward and reasonably priced diffraction grating design was used to present an open-source platform. However, it is only appropriate for experimental remote sensing applications, in which an Unmanned Aerial Vehicle (UAV) should be used to attach the platform. Because it lacks scanning mechanisms, it cannot take pictures while it is motionless [31].

Moreover, understanding the challenges and barriers that affect the acceptance and adoption of OSS is crucial for both developers and organizations. Among the issues raised, technology-related risks and potential barriers in software development occupy a prominent position. This study [7] highlights the most significant risks identified from the perspective of open-source software developers, along with suggested practices for mitigating these risks, providing valuable insights for improving the open-source software development and deployment process. Through a comprehensive literature review (SLR), 14 factors were identified that, from the developers' perspective, represent risks to open-source software (OSS). The most prominent factors that represented challenges were software bugs, lack of product documentation, and poor communication and coordination among developers. Additionally, the researchers conducted a secondary structured review to identify practices that could reduce the impact of these risks in the open-source software environment. This review identified 31 practices aimed at addressing the risk factors related to open-source software development and mitigating their effects.

6.1.3. Bug Tracking, Quality Concerns, and Reporting Mechanisms

According to 60% of respondents (non-developer users) in Raza et al.'s empirical study, poor usability is the primary barrier that OSS applications must overcome to persuade consumers to switch from commercial software [46].

6.1.4. Security and Vulnerability Barriers

The barriers to adopting open-source software include not knowing much about it, people in IT not wanting to change, costs, needing a product that is easy to use and standard, competition from companies that sell their own software, not having enough trained staff, and not receiving good support after buying the software. Additionally, the challenges facing organizations in South Africa in adopting open-source software (OSS) include poor awareness and knowledge of OSS, a lack of specialized technical skills, and limited compatibility of OSS with existing IT infrastructure and traditional applications used in these organizations [17].

Moreover, the technology factor has barriers: total cost of ownership (TCO), reliability, compatibility, complexity, and performance expectancy [17]. No technical obstacles specific to open-source software have been found to apply to OSS-based ERP systems. This was because custom software was common in most ERP setups, and compatibility was not a major concern because ERP systems are mainly used inside a company.

Complexity was also seen as normal for any ERP project, not just open-source ones [17].

In addition, other issues that prevent open-source software from being used in some areas include the lack of skills to develop in-house, limited connections, poor quality, insufficient computer power, and insufficient experts in the field. Other factors, such as removing licenses, not having internal development, insufficient security, and less need for technical help, were considered less important [30].

Furthermore, the socio-technical aspects of software security practices, such as the proficiency of community developers in producing secure code, the caliber of development tools, the degree of testing done prior to product release, and the collaborative practices used throughout the development cycle, determine the trustworthiness of open-source software [48].

As a result, while some developers might overlook the necessity of managing OSS to avoid extra costs, others might not manage them properly because of ignorance or a lack of resources and experience, thus creating security concerns and license breaches [49].

In addition, a class of non-functional requirements (NFRs) pertaining to system availability, confidentiality, and integrity is referred to as security. In many OSS initiatives, this is essential. Experience suggests that addressing security issues, such as lowering information breaches and unauthorized data access, can be achieved by considering security early in the software life cycle [50].

Moreover, threat modeling and other techniques for identifying security requirements are challenging to use in OSS projects. Additionally, during requirements discussions, OSS project stakeholders voice their security concerns. These security issues are typically stated as sentences in comments or external links [50].

Finally, the interests of developers determine how OSS projects are prioritized. Therefore, another crucial challenge in developing an SRE framework for OSS projects is identifying a novel and appropriate way to examine conflicts between various security criteria [50].

6.1.5. ERP-Specific Technical Barriers

Despite the benefits, some respondents voiced concerns about potential security flaws, a lack of official support, and compatibility problems with proprietary systems that could prevent OSS adoption [26].

Moreover, the entire software system lifecycle may be affected by the quality of an issue report. In truth, a lot of software projects routinely delete issue reports that are ambiguous or severely lacking in detail. Because enterprise resource planning (ERP) software systems are complicated and play a vital role in operational organizations, the importance of high-quality issue reports is even more crucial [51].

Zimmermann et al. developed a prototype that assists users in entering the necessary data when reporting a defect and proposed a quality model for bug reports. This study is based on user and developer surveys. The authors' poll reveals a significant discrepancy between what developers think is crucial for bug fixes and what they think is crucial for reporting. However, the developers note that the real issue with fixing a fault is not incorrect information but rather its absence [51].

6.1.6. Developer-Centered Risk Factors and Contribution Dynamics

The most common factors influencing adoption are identified below, while secondary factors are mentioned in point (6.2). [52]

1.Worries about the service and support Provided.
2. Cost / overall cost of owning the product
3.What the product can do and how mature it is.
4. The organization doesn't have enough technical knowledge or skills for open source software.
5. Difficulty of adoption/integration.
6. Viability of the open-source community.
7. Staff knowledge/skills/familiarity with how OSS is different.
8.Worries about ownership and licensing of the software.
9.Worries about the security of the software

In addition, usability experts are not "incentivized by the OSS approach in the way that many hackers are," and they are not "welcomed into OSS projects," according to Nichols and Twidale. These factors are the key reasons why usability experts do not typically participate in OSS projects [47].

6.1.7. Summary of Stakeholder-Reported Usability Challenges

Ensuring the security of the IoT is one of the main challenges it presents. Sensitive information, including what you say and do at home and work, is collected by these devices. Users' faith in the Internet of Things depends on its reliability, but its data security record is poor [53].

Moreover, hackers are increasingly targeting ERP systems to disrupt and steal data from companies because of the increasing number of data breaches and assaults on industries that utilize ERP software, such as manufacturing, hospitality, healthcare, and education. The discovered Tactics, Techniques, Procedures (TTPs) employed by attackers cannot be stopped or detected by conventional ERP application security controls alone. Since many businesses are implementing cloud-based ERP systems, security controls are crucial for responding to changing security threats and safeguarding data availability, confidentiality, and integrity [43].

In addition, the use of mobile devices for access (tablet PC, smartphones, etc.) is another crucial aspect of cloud computing. Mobile devices are becoming increasingly popular because of their ease of use,

which makes them accessible worldwide. Therefore, to increase security, data masking techniques must be applied to the connections between mobile devices and cloud computing [54].

Furthermore, the main security risks to software-intensive systems are software flaws. A software system vulnerability is a feature that can be purposefully or inadvertently exploited to cause a security breach. Static analysis of source code is a method for finding vulnerabilities that examines the written source code and can be used throughout the life cycle without requiring the system to be executable [55].

Additionally, technical vulnerability data on the inspection target are gathered using search engines and open-source information-gathering tools such as Shodan and Censys.

Search engines and social media crawlers are used to gather private information that has been compromised by social engineering assaults [56].

In addition, some businesses worry that because the source code is accessible, hackers may insert back doors into open-source software that might allow unauthorized users to access systems [57].

Moreover, the fact that many researchers do not release their source code or implementations online, particularly in the areas of Privacy-Preserving Data Mining (PPDM) and Privacy-Preserving Utility Mining (PPUM), is a significant issue that slows down research progress on these issues and their industry applications. Therefore, people who wish to use these methods or conduct studies on these subjects must reimplement the algorithms. This requires a considerable amount of time, deep knowledge of data mining and programming abilities, and is error-prone [58].

In addition, it is always possible for each individual or group to produce their own version of the software because open-source software is created by independent developers or groups of developers. If various organizations do not coordinate their efforts, the new features and functionalities they add may not be compatible with one another or show comparable functionality, even if they start with the same source code. Such code changes, known as "forking," caused the open-source BSD-Unix community to split into FreeBSD, OpenBSD, and NetBSD camps in the early 1990s, giving proprietary Microsoft Windows a lead in the operating systems market [59].

Finally, 14 risk factors from the developer perspective were identified based on a SLR study in the context of OSS [7]:

1. Bugs in the source code
2. Lack of adequate product documentation.
3. Lack of effective communication and coordination among developers.
4. Insufficient knowledge (both technical and domain-related).
5. Weak project management practices.
6. Security, intellectual property rights, and legal issues.
7. Inappropriate service support.
8. Compatibility issues.
9. Integration complexities.
10. Lack of user involvement.
11. Lack of testing and quality assurance.
12. Lack of motivation among contributors.
13. Unclear requirements.
14. Lack of funding or financial support.

6.2. Organizational and Structural barriers to OSS adoption

Organizational resistance to change is another significant challenge. Internal reluctance to adopt open-source software often stems from uncertainty about its benefits, fear of disrupting existing workflows, and doubts about long-term support. Many organizations lack strategic planning for open-source software adoption and fail to align their initiatives with broader organizational goals. This lack of clear governance frameworks limits the effective implementation and scaling of open-source software solutions in the health sector. Additionally, weak IT infrastructure can hinder the deployment of open-source software. Organizations with limited resources may struggle to provide the hardware, networking capabilities, or training required to adequately support open-source software platforms. These infrastructure gaps reduce the ease of use of open-source software and discourage its adoption, especially among small businesses and underfunded public institutions.

Moreover, secondary factors that contribute to resistance include [52]**:**

1. Fit for purpose: ability to meet business goals.
2. Complexity: difficulty in implementing or managing.
3. Adherence to standards.
4. Software quality: end-user satisfaction.
5. Software enhancements: innovation over time.

In addition, the use of open-source software depends on the use of commercial software and is influenced by a range of internal and external factors. Successful implementation requires changes in areas such as policies, management, and employee skills and training [52].

Furthermore, the empirical study [17] found that knowledge gaps, not having enough major providers (or vendors), and lower-than-expected costs are some of the reasons why South African organizations find it hard to use open-source ERP systems. The study listed the following factors as the main reasons why organizations in South Africa struggle to adopt OSS ERP systems:

- The organization's level of innovation.
- People who can connect the organization with outside ideas and resources.
- How well the organization has supported

systems in place.

• Whether there is sufficient external support services available.

• How much does the organization know about the product.

The factors and their important parts were divided into two main groups: knowledge problems and not having enough large suppliers. In addition, low costs were found to be a third possible problem that has not been studied sufficiently. The organizational aspect includes barriers related to human and financial resources, the level of innovation, and the role of intermediaries between various organizational units [17].

Based on the findings, support from government agencies and senior leadership, along with factors such as reliability, safety, availability, and ownership costs, were important in helping the adoption process succeed. The paper explains that the biggest influences were cost savings and strong support from top management after looking at several studies from three areas in the public sector: higher education, healthcare, and local government [30].

Furthermore, in the past, many companies did not invest in open-source software because there was insufficient support or training. However, this began to change when large IT companies began supporting open-source software. Companies such as IBM, Hewlett-Packard, and Oracle have all promised to help with Linux using Red Hat and SUSE. This helps suppliers who now offer support for open-source software, and it also makes users feel more confident that they will receive ongoing help [30].

However, the authors point out that the "lack of roadmap" was one of the least significant hazards, which is unexpected considering the high failure rate of OSS initiatives [28]. In addition, facing resistance from their IT department to organizational changes, organizations may lose some important IT competencies [60].

Moreover, Goode identifies the primary causes as management not thinking its products are relevant, worrying about unreliable or temporary support sources, not having enough resources, or not thinking open-source technology is necessary for their companies [35].

Additionally, for mission-critical business operations such as accounting, sales, and manufacturing, most firms rely on outdated systems. These systems, which play crucial operational functions in businesses, were developed several decades ago using antiquated technology that does not work well with modern technologies, is costly to upgrade, and cannot be replaced. Organizations are understandably reluctant to abandon or reconstruct their legacy systems and choose new technologies that can effectively integrate and exchange data with their current legacy systems. Although legacy system integration presents challenges for both proprietary and open-source software, organizations frequently resist using open-source software because of the possibility of insufficient legacy integration and the absence of accountability for unsuccessful integration [59].

Furthermore, adopting open-source systems may necessitate writing off previous expenditures as a "sunk cost," as many firms have already made significant investments in proprietary software before the emergence of open-source software. Naturally, many businesses are unable to implement open-source solutions on an enterprise-wide basis because they are hesitant to pay for this expense. The sunk cost of current proprietary systems makes open-source adoption unjustified, as organizational executives require economic justification for most new technology investments [59].

Finally, the most frequent obstacles to OSS adoption were user opposition, lack of assistance, and shortage of human resources. Additionally, the analysis indicates that the primary motivators for OSS adoption are cost, OSS champions, and top management support [61].

In the public sector, the purchase of IT infrastructure is frequently based on long-term frame contracts with conventional large IT partners, which is another reason why OSS is not (yet) used. Only a small number of these partners have integrated the new OSS proposal into their technical and support teams [38].

6.3. Policy and Ecosystem Constraints

Strategic policy factors and the dynamics of the surrounding environment significantly influence the acceptance of open-source software. The absence of supportive policies, such as government regulations or incentives that encourage the use of OSS, reduces the organizational incentive to migrate from proprietary software to OSS. In the absence of a legal framework that encourages the adoption of open-source software, organizations may find it difficult to embrace it despite its significant benefits.

Moreover, contextual and institutional factors, such as government initiatives, infrastructure availability, and regulatory environments, significantly influence the acceptance of open-source software. While some research highlights the role of social and educational environments in raising awareness of open-source software and encouraging its use [9], other studies emphasize the importance of technical and regulatory conditions that facilitate or hinder adoption [8]. Furthermore, the lack of clear national policies or incentives can weaken institutional motivation to integrate open-source software, especially when policy support is seen as a critical factor for change [15]. In the absence of clear guidelines or enabling frameworks, organizations may be reluctant to abandon proprietary systems despite the potential value of open-source software.

Moreover, the lack of community engagement and collaboration hinders the growth and sustainability of open-source software. The lack of active and influential

communities makes it difficult to build trust and create strong environments around open-source software products, negatively impacting user confidence and long-term sustainability [7].

As noted in [7], sustainability challenges, such as limited financial resources and reliance on donations, prohibit the long-term viability of many commercial open-source projects. This uncertain situation undermines user confidence, as projects that consistently support them are liquidated and pose a greater risk to those who rely on them.

Furthermore, legal issues surrounding licensing further complicate the situation. The ambiguity surrounding open-source software licenses, coupled with concerns about intellectual property rights, may prompt organizations to avoid using this software to prevent legal violations or litigation. This highlights the need for clear laws and training on open-source software licenses [7].

In addition, environmental barriers that may hinder adoption include the necessary technical expertise and technical support for the product, issues of credibility and legitimacy, and the need for accessible external support services. Furthermore, the long-term environmental stability of the platform is an important factor, and the level of user familiarity with the product also plays a role [17].

IoT device security protocols are not standardized. In the absence of a consistent taxonomy, guaranteeing the security of an IoT ecosystem is challenging. which includes various devices, apps, and communication protocols [53].

In addition, 64% of enterprises still consider security a significant barrier to the use of OSS [28].

Moreover, because of security concerns and malevolent intent from both inside and outside the company, organizations are hesitant to use cloud ERP. To address the security-related problems with ERP, this study uses a literature review to identify the different attacks that an ERP system is vulnerable to, determines what security controls should be implemented to create a more secure environment while considering NIST 800-53 R5 and ISO/IEC 27001:2013, and maps the identified security controls to help Canada comply with PIPEDA [43].

Additionally, a hybrid solution has been suggested as an efficient way for businesses to enjoy the advantages of cloud ERP while maintaining sensitive and private applications on-site. When sensitive data are hosted by external cloud service providers, there are two primary data security concerns regarding confidentiality: uncertainty about how the data are stored and a lack of power and control over security policies and standards [43].

In addition, open-source tools are least likely to be used by businesses in sectors such as healthcare and finance that have legal security obligations. Rather, companies will likely continue to rely on suppliers who can be held accountable for security lapses [57].

However, if OSS is used carelessly, there are serious legal and security issues that could endanger end users' privacy and security and result in large financial losses for developers and businesses [49].

Despite this, many governments have put laws in place to encourage the adoption of open-source software (OSS) in public agencies. However, recent research has revealed that this adoption rate is low and that OSS alternatives have not been used to replace proprietary software. The causes may differ based on the elements that could cause this decline in OSS adoption, such as whether the government encourages or mandates the policy for public entities, a lack of trust, and the absence of expert assistance [61].

According to a survey, OSS acceptance remains low in Malaysian public sector firms, and its implementation is limited due to a lack of specific elements. By elucidating the current situation of open-source software adoption in Malaysia and addressing any influencing or impeding factors [61].

Furthermore, the Fitzgerald framework identifies four factors that could influence OSS adoption. This comprises organizational-level support factors, subjective norms, and managerial interventions. However, the only element that appears to have an impact on OSS adoption at the national level is uncertainty avoidance [61].

Finally, the goals of public sector policies are either to promote OSS over proprietary software in public procurement or to increase proficiency with Open Source and maximize outcomes within the public sector. These policies vary in their scope, methods of implementation, and degree of prescriptiveness, from enforceable laws to straightforward norms [41].

7. RESULT AND DISCUSSION

Following a systematic selection of 61 key studies, a thorough evaluation was conducted to identify the drivers of open-source software adoption. The data reveal a complex interplay between technical, institutional, and personal factors, demonstrating that adoption is rarely a straightforward process. While established behavioral approaches, such as the Technology Acceptance Model (TAM), the Unified Theory of Acceptance and Use of Technology (UTAUT) Model, and Self-Determination Theory (SDT), provide strong theoretical support, their proactive power varies considerably depending on the institutional environment. These findings address research inquiries by demonstrating the diverse factors influencing free software licensing and highlighting the need for context-specific measures. **Figure 2** introduces an integrated conceptual framework that maps the primary categories from the 61 reviewed studies.

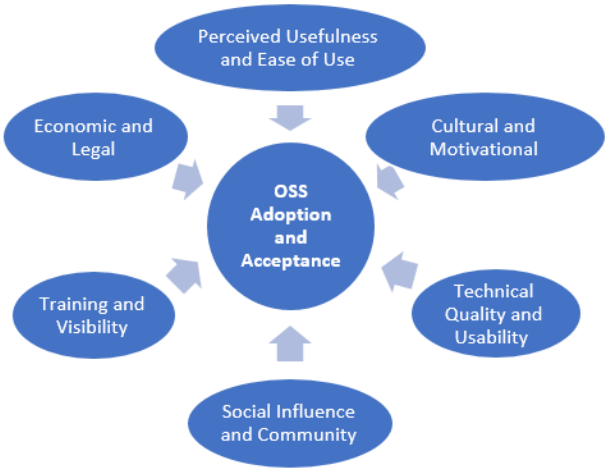

Figure 2. The integrated conceptual framework for OSS adoption and acceptance factors.

7.1. Discussion of Comparative Findings and Contradictions

A careful assessment of the data revealed that the motivations for adoption were not uniform across all settings. The motivation for an individual to use open-source software in a university setting may differ significantly from that in a government department or private institution. For example, Silva et al. (2023) and Tome et al. (2014) agree on the need for technical support, but differ on the primary obstacle: Silva emphasizes external compatibility, while Tome highlights the internal shortcomings of qualified personnel within organizations.

In addition, the effect of “perceived ease of use” (from the technology acceptance model) shows mixed results; it is a key factor in educational settings [9], but appears to be less central in specialized e-government security assessments [54], where “system quality” and “Trust” are of utmost importance.

These variations highlight the vulnerability of applying standard behavioral approaches without adapting them to the local context, underscoring the imperative of understanding local technical, organizational, and social realities. Because of these differences, it becomes clear that technology adoption is strongly linked to the specific governance environment and cannot be achieved in isolation.A key finding is the importance of adjusting acceptance frameworks to suit specific circumstances. Standardized models often fail because they aim to provide a single, comprehensive solution, overlooking the specific details and circumstances of each sector. This underscores the critical need for tailored frameworks—models built from the ground up to consider the unique needs and social contexts of each area. The impact of organizational culture and management controls on adaptation outcomes is significant, highlighting that technology adoption is deeply intertwined with the economic and social environment and cannot be considered in isolation from it.

7.2. Defining the Review’s Contribution

This review goes beyond current descriptive summaries by integrating diverse variables into a single framework that combines assessments of technical, social, and personal capabilities (Figure 2). Unlike previous research, which often focused on a single dimension (such as the technical or psychological dimension specifically), the significance of this research lies in highlighting the interaction between these dimensions. Unlike previous systematic reviews that addressed technical, social, or personal factors in a fragmented manner, this framework elucidates the interconnections between these dimensions. This approach not only lists the elements but also fills a crucial gap by including social and personal characteristics that previous frameworks have overlooked.

7.3. Implications for the Yemeni Context & Policy Interventions

Given the exceptional economic and structural challenges facing Yemen, this systematic assessment examines the data and provides a pivotal guiding framework for the country’s technological transformation. Owing to financial constraints and the limited IT infrastructure in Yemen, the following regulatory actions are directly derived from the TOEI-individual framework to ensure the successful adoption of open-source software solutions. In the Yemeni government sector, where the high cost of proprietary software represents a constant strain on financial resources, the use of open-source software is no longer merely a technical option but a fundamental requirement for the government. Policies should concentrate on institutional support and operational ease to guarantee the compatibility of these systems with the local administrative context. Based on this holistic framework, the following regulatory actions are proposed to effect real change.

1. **Building Institutional Capacity:**
Government agencies should establish "temporary open-source software committees" to examine current software dependencies and identify free-source alternatives. These initiatives should also include educational programs that go beyond basic technical training. Such programs should leverage "self-motivation" (as outlined in the SDT-TAM model) to foster confidence among public sector employees, thereby reducing resistance to the transition.
2. **Prioritizing Local Usability & Standards:** Policy initiatives should prioritize local accessibility. Given the constraints of Yemeni administrative systems, a one-size-fits-all approach is not feasible. To ensure long-

term sustainability, these initiatives must provide robust support for the Arabic language and a user-friendly interface specifically designed to meet the everyday needs of Yemeni users.

3. **Legislative Support & Digital Sovereignty**
Priority guidelines for open-source software must be established to promote digital sovereignty in government procurement processes. At the legislative level, national plans and support measures are crucial for overcoming structural obstacles. Governments and regulators must adopt incentive strategies that facilitate open software adoption, streamline funding, and remove legal barriers to support the resilience and sustainability of open software ecosystems. It is not just about adopting software; it is about fostering local skills. This includes providing smart incentives for Yemeni programmers to take ownership—not just of the code, but also of the adaptation and long-term sustainability that national bodies require to succeed.

4. **Academic-Government Synergy:**
Universities in Yemen must align their IT curricula with open-source software principles. Educational institutions play a crucial role in this endeavor. By directly integrating open-source principles into their programs, they not only impart technical skills but also contribute to cultivating a cadre of digital leaders with the intellectual capacity to drive innovation in open-source software across the country. These efforts support the development of students' capabilities and foster positive attitudes from the earliest stages of their education, helping them overcome the technical obstacles that may hinder the adoption of open-source software.

8. CONCLUSION AND FUTURE RESEARCH

This study systematically interpreted 61 pivotal studies through a systematic review of available works to identify the key factors influencing the adoption of open-source software and answer the research questions. This study successfully addressed these questions by proposing a comprehensive framework and outlining the necessary strategic measures for the Yemeni context.

This study examines the key factors influencing how individuals accept and use open-source software through a review and evaluation of existing research in this field. The study employs several theoretical approaches, including the Technology Acceptance Model (TAM), Unified Theory of Acceptance and Use of Technology (UTAUT), and Self-Determination Theory (SDT), with results ranked according to the TOEI methodology. Based on the reviewed research, there is a consensus on several elements, such as perceived usefulness, ease of use, training, visibility, cultural motivation, and system quality. It is important to note that the evaluation reveals that while technical and physical aspects are of paramount importance, psychological, social, and cultural components, such as social identity, also play a pivotal role, particularly in educational or administrative settings in developing countries.

This systematic review yielded several notable findings regarding the influences on open-source software adoption while also revealing several key research gaps. First, there is a clear lack of empirical research investigating the impact of cultural, social, and motivational factors on the adoption of open-source software, particularly in educational and governmental contexts in developing nations. Second, the limited understanding of the interplay between psychological, social, and technological elements hinders the development of accurate and comprehensive models to explain this behavior. Third, the proposed integrated architecture (TOEI-individual) has not yet been experimentally tested in specific local contexts, which limits the generalizability of the findings. Fourth, most current studies are limited to short time periods (sectoral studies) and do not provide insights into how individuals' perceptions of open-source software change over time or after acquiring knowledge and skills. Fifth, the absence of standard quantitative complementary analyses prevented a comprehensive assessment of the impact of various variables. Finally, there is a lack of systematic comparisons between proprietary and open-source software in various areas, which weakens the understanding of the actual features and difficulties of the software.

Based on the identified research gaps, it is suggested that future research should focus on testing and applying the standardized TOEI-individual model within the Yemeni context, particularly in the government sector and academic institutions, to assess its effectiveness in explaining local free software adoption trends. It is important to conduct empirical studies that incorporate technical, psychological, social, and cultural components to understand the complex interplay between them and their impact on users' intentions and actions. Ongoing investigations are also recommended to monitor the growth of individuals' perceptions of free software as they gain experience and participate in training programs. Furthermore, standardized meta-assessments can be used when more consistent data are available to estimate the strength of disparate components and conduct systematic comparisons between proprietary and free software across various fields. Finally, we recommend investigating the impact of administrative constraints and local needs on the ease of distributing free software to ensure a smooth and sustainable transition from proprietary to free software while providing practical guidance for national legislation and e-governance in Yemen.

This review makes a valuable contribution to the literature by moving beyond isolated studies to offer a comprehensive six-dimensional conceptual framework that integrates individual psychological motivations with

structural organizational determinants. In total, it synthesizes the findings of 61 diverse studies and provides a comprehensive roadmap for understanding the adoption of open-source software in complex environments. Finally, this review contributes to theory by offering a structured, concept-centric synthesis across disciplines and to practice by highlighting the factors that require strategic attention when planning OSS implementation.

REFERENCES

[1] D. G. Silva, C. Coutinho, and C. J. Costa, "Factors influencing free and open-source software adoption in developing countries—an empirical study," *J. Open Innov. Technol. Mark. Complex.*, vol. 9, no. 1, Mar. 2023. DOI: 10.1016/j.joitmc.2023.01.002.

[2] F. Gustavo, S. Gripe, and I. A. Rodello, *A brief survey of open source erp systems usage on brazilian organizations*, [Online], 2012. [Online]. Available: http://www.opensource.org.

[3] T. Mladenova, "Open-source erp systems: An overview," in *2020 International Conference Automatics and Informatics (ICAI)*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020. DOI: 10.1109/ICAI50593.2020.9311331.

[4] D. Petrov, *Adoption barriers of open-source software: A systematic review*, [Online], 2018. [Online]. Available: https://ssrn.com/abstract=3138085.

[5] R. A. Ebardo, *Visibility and training in open source software adoption: A case in philippine higher education*, 2018.

[6] M. R. Al-Hajri, M. G. Al-Mukhaini, and M. R. Ramalingam, *Adoption of free and open source software using alternative educational framework in college of applied sciences*, Feb. 2017.

[7] S. Haider, W. Khalil, A. S. Al-Shamayleh, A. Akhunzada, and A. Gani, "Risk factors and practices for the development of open source software from developers' perspective," *IEEE Access*, vol. 11, pp. 63 333–63 350, 2023. DOI: 10.1109/ACCESS.2023.3267048.

[8] T. A. Alrawashdeh, M. W. Elbes, A. Almomani, F. ElQirem, and A. Tamimi, "User acceptance model of open source software: An integrated model of oss characteristics and utaut," *J. Ambient Intell. Humaniz. Comput.*, vol. 11, no. 8, pp. 3315–3327, Aug. 2020. DOI: 10.1007/s12652-019-01524-7.

[9] F. J. Racero, S. Bueno, and M. D. Gallego, "Predicting students' behavioral intention to use open source software: A combined view of the technology acceptance model and self-determination theory," *Appl. Sci.*, vol. 10, no. 8, Apr. 2020. DOI: 10.3390/APP10082711.

[10] K. A. Dawood, K. Y. Sharif, A. A. Zaidan, A. A. A. Ghani, H. B. Zulzalil, and B. B. Zaidan, "Mapping and analysis of open source software (oss) usability for sustainable oss product," *IEEE Access*, vol. 7, pp. 65 913–65 933, 2019. DOI: 10.1109/ACCESS.2019.2914368.

[11] K. A. Dawood, K. Y. Sharif, A. A. Ghani, H. Zulzalil, A. A. Zaidan, and B. B. Zaidan, "Towards a unified criteria model for usability evaluation in the context of open source software based on a fuzzy delphi method," *Inf. Softw. Technol.*, vol. 130, Feb. 2021. DOI: 10.1016/j.infsof.2020.106453.

[12] E. Kadasah and O. Alrwais, *Evaluation of training modules in open source erp*, [Online], 2022. [Online]. Available: https://www.researchgate.net/publication/361326041.

[13] K. L. Gwebu and J. Wang, "Adoption of open source software: The role of social identification," *Decis. Support Syst.*, vol. 51, no. 1, pp. 220–229, Apr. 2011. DOI: 10.1016/j.dss.2010.12.010.

[14] M. D. Gallego, S. Bueno, F. J. Racero, and J. Noyes, "Open source software: The effects of training on acceptance," *Comput. Hum. Behav.*, vol. 49, pp. 390–399, Aug. 2015. DOI: 10.1016/j.chb.2015.03.029.

[15] V. Bhatiasevi and D. Krairit, "Acceptance of open source software amongst thai users: An integrated model approach," *Inf. Dev.*, vol. 29, no. 4, pp. 349–366, Nov. 2013. DOI: 10.1177/0266666912465880.

[16] S. W. V. Rooij, "Open source software in us higher education: Reality or illusion?" *Educ. Inf. Technol.*, vol. 12, no. 4, pp. 191–209, Dec. 2007. DOI: 10.1007/s10639-007-9044-6.

[17] L. Tome, K. A. Johnston, A. Meadows, and K. Allan, *Barriers to open source erp adoption in south africa*, [Online], 2014. [Online]. Available: https://digitalcommons.kennesaw.edu/ajis/vol6/iss2/1.

[18] J. Linåker, H. Munir, K. Wnuk, and C. E. Mols, "Motivating the contributions: An open innovation perspective on what to share as open source software," *J. Syst. Softw.*, vol. 135, pp. 17–36, Jan. 2018. DOI: 10.1016/j.jss.2017.09.032.

[19] M. D. Gallego, P. Luna, and S. Bueno, "User acceptance model of open source software," *Comput. Hum. Behav.*, vol. 24, no. 5, pp. 2199–2216, Sep. 2008. DOI: 10.1016/j.chb.2007.10.006.

[20] V. Midha and P. Palvia, "Factors affecting the success of open source software," *J. Syst. Softw.*, vol. 85, no. 4, pp. 895–905, Apr. 2012. DOI: 10.1016/j.jss.2011.11.010.

[21] K. Ven and J. Verelst, *The impact of ideology on the organizational adoption of open-source software*.

[22] E. Katsamakas and M. Xin, "Open source adoption strategy," *Electron. Commer. Res. Appl.*, vol. 36, Jul. 2019. DOI: 10.1016/j.elerap.2019.100872.

[23] V. Lenarduzzi, D. Taibi, D. Tosi, L. Lavazza, and S. Morasca, "Open source software evaluation, selection, and adoption: A systematic literature review," in *Proceedings of the 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA 2020)*, Institute of Electrical and Electronics Engineers Inc., Aug. 2020, pp. 437–444. DOI: 10.1109/SEAA51224.2020.00076.

[24] A. Zhussupova and A. A. Rahman, *Open Source Software Adoption in Public Organizations of Kazakhstan*. IEEE, 2011.

[25] M. A. Uddin, M. S. Alam, A. A. Mamun, T. U. Z. Khan, and A. Akter, "A study of the adoption and implementation of enterprise resource planning (erp): Identification of moderators and mediator," *J. Open Innov. Technol. Mark. Complex.*, vol. 6, no. 1, Mar. 2019. DOI: 10.3390/JOITMC6010002.

[26] A. M. Taha, A. A. Abbood, A. A. A. Razzaq, and A. S. Al-Bahri, "Identifying the affecting factors for adoption of open source software in it community," *J. Eng. Appl. Sci.*, vol. 13, no. 14, pp. 5771–5780, 2018. DOI: 10.3923/jeasci.2018.5771.5780.

[27] G. L. Kovács, S. Drozdik, P. Zuliani, and G. Succi, *Open source software for the public administration*, 2004.

[28] M. Silic and A. Back, *Identification and importance of the technological risks of open source software in the enterprise adoption context*, [Online], 2015. [Online]. Available: http://aisel.aisnet.org/wi2015.

[29] A. Raza, L. F. Capretz, and F. Ahmed, "An empirical study of open source software usability: The industrial perspective," *Int. J. Open Source Softw. Process.*, vol. 3, no. 1, pp. 1–16, Jan. 2011. DOI: 10.4018/jossp.2011010101.

[30] M. G. Yaseen, S. A. Abd, and I. Adeb, "Critical factors affecting the adoption of open source software in public organizations," *Iraqi J. for Comput. Sci. Math.*, vol. 1, no. 2, pp. 29–36, 2020. DOI: 10.30880/ijcsm.0000.00.00.000.

[31] A. Al-Hourani, S. Balendhran, S. Walia, and T. Hourani, "Line scan hyperspectral imaging framework for open source low-cost platforms," *Remote. Sens.*, vol. 15, no. 11, Jun. 2023. DOI: 10.3390/rs15112787.

[32] A. Raza, L. F. Capretz, and F. Ahmed, "Users' perception of open source usability: An empirical study," *Eng. Comput.*, vol. 28, no. 2, pp. 109–121, Apr. 2012. DOI: 10.1007/s00366-011-0222-1.

[33] A. Raza and L. F. Capretz, "Contributors preference in open source software usability: An empirical study," *Int. J. Softw. Eng. Appl.*, vol. 1, no. 2, pp. 45–64, Apr. 2010. DOI: 10.5121/ijsea.2010.1204.

[34] M. Alshammari, R. Anane, and R. J. Hendle, "An e-learning investigation into learning style adaptivity," in *Proceedings of the Annual Hawaii International Conference on System Sciences (HICSS)*, IEEE Computer Society, Mar. 2015, pp. 11–20. DOI: 10.1109/HICSS.2015.13.

[35] B. Johansson and F. Sudzina, *Choosing open source erp systems: What reasons are there for doing so?* 2009.

[36] S. Lakka, T. Stamati, C. Michalakelis, and D. Anagnostopoulos, "Cross-national analysis of the relation of egovernment maturity and oss growth," *Technol. Forecast. Soc. Chang.*, vol. 99, pp. 132–147, Oct. 2015. DOI: 10.1016/j.techfore.2015.06.024.

[37] C. Payne, *On the security of open source software*, 2002.

[38] P.-E. Schmitz, *Study into the use of open source software in the public sector: The open source market structure*, A report directed by an IDA Study Interchange of Data between Administrations, European Commission, DG Enterprise, 2001.

[39] V. R. Sánchez, P. N. Ayuso, J. A. Galindo, and D. Benavides, "Open source adoption factors–a systematic literature review," *IEEE Access*, 2020. DOI: 10.1109/ACCESS.2020.2993248.

[40] M. F. Baharuddin, T. A. T. Izhar, and M. S. M. Shoid, "Adoption of open source software (oss) and organization performance in the library," *Int. J. Acad. Res. Bus. Soc. Sci.*, vol. 8, no. 9, Oct. 2018. DOI: 10.6007/ijarbss/v8-i9/4591.

[41] P. Grzegorzewska, A. Katz, S. Muto, S. Pätsch, and T. Schubert, *The impact of open source software and hardware on technological independence, competitiveness and innovation in the eu economy: Final study report*, 2020.

[42] P. Saa, A. C. Costales, O. Moscoso-Zea, and S. Lujan-Mora, "Moving erp systems to the cloud - data security issues," *J. Inf. Syst. Eng. Manag.*, vol. 2, no. 4, Dec. 2017. DOI: 10.20897/jisem.201721.

[43] P. Gottipati, B. Swar, and P. Zavarsky, "Information security considerations for cloud-based erp system and best practices for its retirement phase," *Int. J. Comput. Appl. (IJCA)*, 2020.

[44] F. Jansen, S. Jansen, and F. Hou, "Trustseco: An interview survey into software trust," Jan. 2021, Online. [Online]. Available: http://arxiv.org/abs/2101.06138.

[45] B. B. Chua and D. V. Bernardo, "Open source developer download tiers: A survival framework," IEEE, 2013.

[46] L. Llerena, N. Rodriguez, J. W. Castro, and S. T. Acuña, "Adapting usability techniques for application in open source software: A multiple case study," *Inf. Softw. Technol.*, vol. 107, pp. 48–64, Mar. 2019. DOI: 10.1016/j.infsof.2018.10.011.

[47] A. Raza, L. F. Capretz, and F. Ahmed, "Improvement of open source software usability: An empirical evaluation from developers' perspective," *Adv. Softw. Eng.*, vol. 2010, pp. 1–12, Sep. 2010. DOI: 10.1155/2010/517532.

[48] S.-F. Wen, "Software security in open source development: A systematic literature review," 2017.

[49] R. Duan, "Toward solving the security risks of open-source software use," 2019.

[50] W. Wang, "Towards a security requirements management framework for open-source software," in *Proceedings - 2018 IEEE 26th International Requirements Engineering Conference (RE 2018)*, Institute of Electrical and Electronics Engineers Inc., Oct. 2018, pp. 478–483. DOI: 10.1109/RE.2018.00065.

[51] L. Aversano, "Issue reports analysis in enterprise open source systems," in *ICEIS 2019 - Proceedings of the 21st International Conference on Enterprise Information Systems*, SciTePress, 2019, pp. 337–344. DOI: 10.5220/0007757803370344.

[52] B. Buffett, "Factors influencing open source software adoption in public sector national and international statistical organisations," Feb. 2014.

[53] U. Tariq, I. Ahmed, A. K. Bashir, and K. Shaukat, "A critical cybersecurity analysis and future research directions for the internet of things: A comprehensive review," *MDPI*, Apr. 2023. DOI: 10.3390/s23084117.

[54] M. Yesilyurt and Y. Yalman, "New approach for ensuring cloud computing security: Using data hiding methods," *Sadhana - Acad. Proc. Eng. Sci.*, vol. 41, no. 11, pp. 1289–1298, Nov. 2016. DOI: 10.1007/s12046-016-0558-8.

[55] A. Nguyen-Duc, M. V. Do, Q. L. Hong, K. N. Khac, and A. N. Quang, "On the adoption of static analysis for software security assessment–a case study of an open-source e-government project," *Comput. & Secur.*, vol. 111, Dec. 2021. DOI: 10.1016/j.cose.2021.102470.

[56] S. Lee and T. Shon, "Open source intelligence base cyber threat inspection framework for critical infrastructures," IEEE, 2016.

[57] G. Lawton, "Open source security opportunity or oxymoron," Mar. 2002.

[58] J. C. W. Lin, P. Fournier-Viger, L. Wu, W. Gan, Y. Djenouri, and J. Zhang, "Ppsf: An open-source privacy-preserving and security mining framework," in *IEEE International Conference on Data Mining Workshops (ICDMW)*, IEEE Computer Society, Jul. 2018, pp. 1459–1463. DOI: 10.1109/ICDMW.2018.00208.

[59] D. Nagy, A. M. Yassin, and A. Bhattacherjee, "Organizational adoption of open source software: Barriers and remedies," *Commun. ACM*, vol. 53, no. 3, pp. 148–151, Mar. 2010. DOI: 10.1145/1666420.1666457.

[60] J. Duan, P. Faker, A. Fesak, and T. Stuart, *Benefits and drawbacks of cloud-based versus traditional erp systems*, 2012.

[61] M. G. Yaseen and M. Bahari, *A theoretical research framework of open source software adoption in malaysian university information and communications technology centers*, 2014.