Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2023) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

CENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies 2023

A Product-Line-Engineering Framework for Secure Enterprise-Resource-Planning Systems

Richard May[a,*], Christian Biermann[a], Andy Kenner[b], Jacob Krüger[c], Thomas Leich[a]

[a] *Harz University Wernigerode, Germany*
[b] *Otto-von-Guericke University Magdeburg, Germany*
[c] *Eindhoven University of Technology, The Netherlands*

Abstract

Enterprise-resource-planning (ERP) systems are highly complex, incorporating critical data and configuration options that can easily cause security threats or risks. While product-line engineering (PLE) provides methods for dealing with configurability, security concerns are not of primary concern in established PLE frameworks. In this paper, we extend the perspectives of established PLE frameworks to incorporate security engineering for ERP systems. We build on a literature review of 15 years (2008–2022) of research on security engineering for ERP systems or product lines. Our framework incorporates three perspectives (i.e., application, security, and domain engineering) with 16 processes and 24 activities. We discuss our framework with respect to the ERP reference architecture and relevant security concerns. Our contributions are intended to help researchers and practitioners obtain a reference on how security engineering can be integrated into PLE for ERP systems.

Keywords: enterprise resource planning; product-line engineering; software engineering; security

1. Introduction

Enterprise-resource-planning (ERP) systems help companies define, monitor, and control their business processes efficiently [23], which is why the importance of and dependence on ERP systems is constantly growing [9]. An ERP system is usually based on a set of basic features extended with customer-specific ones, resulting in a number of similar, yet customized, system variants. As a consequence, ERP systems build on variability mechanisms to configure the

* Corresponding author. Tel.: +49 3943 659 300.
E-mail address: rmay@hs-harz.de

system's features for individual companies, business processes, or use cases [4, 29]. For this reason, we refer to ERP systems as highly-configurable systems, typically developed using product-line engineering (PLE) techniques [5]. Due to the complexity of such systems and the increased use of cloud services [38], there is a high risk of cyber attacks that can cause data breaches [1, 2] by exploiting system vulnerabilities that occur frequently in highly-configurable systems, for example, because of unknown feature interactions or complex configuration options [13, 33].

There is extensive research (cf. Section 6) on the intersections between PLE and security [15, 27] as well as between engineering ERP systems and PLE [4]. However, the intersection of all three, specifically using PLE to develop secure ERP systems, has not received much attention in research. So, corresponding tool support is not mature, which is why ERP systems are usually developed without considering security and configuration options in combination. We argue that existing frameworks from any of the three domains (i.e., PLE, security, ERP systems) are only partially applicable to this intersection or require significant extensions. **We aim to tackle this gap by contributing a PLE framework for developing secure ERP systems**. To achieve our research goal, we combined an empirical-to-conceptual (i.e., literature analysis) and conceptual-to-empirical methodology (i.e., author experiences and discussions) according to Nickerson et al. [35] to synthesize a unified framework. We also map the ERP reference architecture and security concerns onto our framework to provide practitioners and researchers a guide for developing highly-configurable, yet secure, ERP systems.

2. Background

In this section, we provide background information on the key concepts of ERP systems and PLE.

2.1. ERP Systems

An ERP system is an integrated software system that represents, coordinates, and monitors business processes in an automated and unified way [7, 41]. ERP systems implement various internal and external features, such as planning, monitoring, and controlling resources or finances [39], even involving external parties (e.g., suppliers, customers) [4]. The features of an ERP system are implemented via integrated modules (e.g., billing) that are customized to the requirements of a customer. Consequently, such modules are usually oriented towards customer demands, industry standards, or regulations and laws [29].

Typically, ERP systems rely on six layers as their reference architecture: (1) *Business processes* represent the value for a company, involving all features and information of the ERP system (e.g., for controlling finance data) [14]. (2) *User interfaces* enable humans to use an ERP system and can be implemented in various forms, for instance, as a web application [6]. (3) *Transactions* are an essential pillar of an ERP system, enabling reusable data processing and data exchange within the business processes [40]. (4) *Interfaces* refer to software that is essential for processing and exchanging data, for example., the IDoc format for data exchange [36]. (5) *Storage* is related to the underlying medium or technology for storing and processing data, such as an SQL database in a cloud environment [27, 37]. (6) *Hardware* mainly refers to actuators, sensors, or network components and their integration into the ERP system, for instance, as Internet-of-Things applications [16, 42]. These layers and their dependencies are complex, leading to various attack vectors, such as SQL injection [11]. Thus, potential attack vectors for every layer, suitable security measures, and security goals in accordance to (domain-specific) security standards must be considered when developing secure ERP systems [25].

2.2. Configurable Systems and PLE

A configurable software system (e.g., the Linux Kernel) is an integrated platform that implements reusable features (i.e., user-visible functionalities) that can be configured (e.g., enabling, disabling, setting to a specific value) to customize individual variants of the platform [5]. Typically, such systems employ PLE concepts, such as feature modeling [34], to manage the platform, implement its configuration options via a variability mechanism, and automatically derive as well as test variants. PLE enables organizations to efficiently engineer a large portfolio of similar variants based on the integrated platform, leading to higher quality, faster delivery, and facilitated maintenance [21, 24].

PLE is typically oriented towards two perspectives and their corresponding processes [5]. In the *domain engineering*, the integrated platform itself is designed, including tasks like eliciting domain requirements, modeling features,

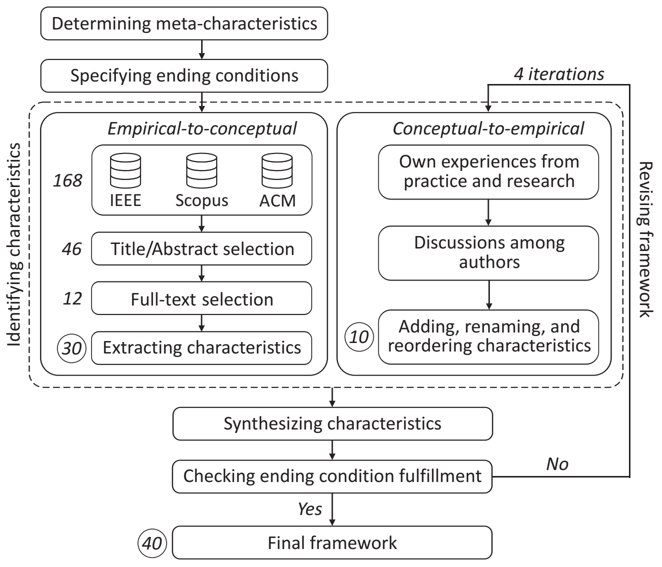
Determining meta-characteristics
Specifying ending conditions
4 iterations
Identifying characteristics
Empirical-to-conceptual
168
IEEE
Scopus
ACM
46
Title/Abstract selection
12
Full-text selection
30
Extracting characteristics
Conceptual-to-empirical
Own experiences from practice and research
Discussions among authors
10
Adding, renaming, and reordering characteristics
Revising framework
Synthesizing characteristics
Checking ending condition fulfillment
No
Yes
40
Final framework

Fig. 1. Method for developing our framework. Numbers indicate the amount of papers or characteristics (circled), namely processes and activities.

designing the platform architecture, and implementing configuration options. In the *application engineering*, individual variants of the platform are implemented by mapping concrete customer requirements to the features of the platform to derive, adapt (e.g., to hardware constraints), and test a variant. Despite changes in how PLE processes are executed in modern software engineering and less focus on these two perspectives [22], they remain key to PLE.

3. Framework Development

To systematically construct our framework, we built on the guidelines by Nickerson et al. [35] (cf. Figure 1). Originally, these guidelines refer to taxonomy development. However, the terms taxonomy and framework can be referred to as interchangeably in our context [35], meaning that the method is appropriate for our work.

Determining Meta-Characteristics. First, we defined meta-characteristics of our framework, which scope the data extraction for processes and activities related to ERP systems, PLE, and security. In our case, these characteristics refer to the perspectives of domain engineering, security engineering, and application engineering [31]. Orthogonal to these perspectives, we distinguish between the problem space (i.e., domain abstractions like requirements), solution space (i.e., implementation of these abstractions), and the mapping between the two (i.e., the configuration options) [5].

Specifying Ending Conditions. Ending conditions (E_X) specify quality criteria that shall hold at the end of the framework-development process [35]. From the *objective perspective*, the framework involves unique characteristics only **(E_1)**, no information is taken out of context **(E_2)**, and no characteristics are merged or added in the last iteration **(E_3, E_4)**. From the *subjective perspective*, the framework shall be robust **(E_5)**, comprehensive **(E_6)**, extendable **(E_7)**, and explanatory **(E_8)**. We do not consider other conditions, since our framework is not employed in the real world, yet.

Identifying Characteristics. Since sufficient knowledge for developing our framework is available in the literature (e.g., secure frameworks), we decided to employ an empirical-to-conceptual construction strategy [35]. Precisely, we based our framework on a qualitative literature review [18], focusing on secure frameworks for ERP or PLE-based systems—instead of analyzing papers that describe use-case-specific engineering processes for certain systems. This way, we aimed to abstract, synthesize, and extend the processes and activities of existing frameworks to create a new, more comprehensive framework for our specific case.

First, we defined the following search string: `"secur*" AND "framework" AND (("ERP" OR "enterprise resource planning") OR ("product line"))`. Second, we defined the selection criteria, including only peer-reviewed conference or journal papers with more than four pages that have been published in the last 15 years (2008–2022). We excluded frameworks that focus on detailed activities (e.g., auditing, certification) to achieve our research goal of defining a consolidated and comprehensible framework. The first and second authors conducted the automated search on January 15, 2023, resulting in 168 papers (Scopus: 116; IEEE Xplore: 30; ACM Guide to Computing

Literature: 22). After reading titles and abstracts, we kept 46 papers as potentially relevant. In the end, we included 12 of those 46 papers as relevant after reading their full-texts [3, 8, 9, 10, 12, 17, 30, 31, 32, 43, 44, 45].
Synthesizing Characteristics. The first and second authors continued to abstract and synthesize the perspectives, processes, and activities of the identified frameworks according to our meta-characteristics. This resulted in a first version of the framework with 13 processes and 17 activities. We then discussed this first iteration among the first three authors and evaluated it with respect to our ending conditions. Not surprisingly, we did not meet E_1, E_3, E_4, and E_5 at this point.
Revising the Framework. To revise our framework, we employed a conceptual-to-empirical construction strategy [35], relying on our practical and research experiences in security, ERP systems, and PLE [15, 20, 21, 22, 15, 26, 27, 28]. So, we incrementally added, renamed, or reordered ten processes and activities by using an open-card sorting-like method until we met all ending conditions. After four iterations, our final framework includes three perspectives, 16 processes, and 24 activities (cf. Figure 2).

4. PLE Framework for Secure ERP Systems

Next, we describe our framework from the engineering perspective of a secure ERP product line (cf. Figure 2). We remark that the processes can be executed from both top to bottom and bottom to top (i.e., for re-engineering [19, 22]).

4.1. Domain Engineering

Problem Space. The domain engineering starts with *product line scoping*, which includes obtaining a *domain understanding* and *business understanding* for the ERP product line. This first process covers all factors that influence the success of the product line within the domain (e.g., industry), ideally involving all relevant stakeholders (e.g., customers, developers) [3, 17]. In addition, these understandings can prevent design misfits, so that the ERP product line meets the expectations of its customers [44, 45]. The *product line requirements analysis* consists of four activities. Referring to ERP systems, it is essential to know their *reference architecture*, its associated requirements (e.g., usability of the user interface) [3, 17], and all relevant *business processes* (e.g., controlling) [12]. Moreover, reusable *common requirements* (e.g., fundamental features based on the domain understanding) [3, 10, 30] and *variable requirements* (i.e., configurable features based on customer demands) should be considered in this analysis [3, 30].
Mapping. To map problem and solution space, *product line modeling and design* must be executed. This process mainly refers to decisions on the architectural level of the ERP product line. Usually, a variability model (e.g., feature model [34]) is used to represent features, their relationships, and dependencies [3, 30, 31, 43].
Solution Space. The solution space involves the *domain realization* and *domain testing*. These processes refer to a detailed implementation plan, which includes the design of reusable features, policies, and available services according to the design specifications. Finally, the results of all domain-engineering processes are validated and verified based on the implemented assets. Optimally, early and frequent testing is established to support an efficient testing process based on reusable test artifacts (e.g., test data, metrics) [3, 30].

4.2. Security Engineering

Problem Space. The first security-engineering activities are related to a *security analysis* [32, 43]. Specifically, these activities are the review and documentation of *domain requirements* (e.g., domain-specific interfaces) [17, 30], *domain legal regulations* (e.g., policies, standards, laws) [8, 30, 31, 32, 45], *third party services* (e.g., supplier systems, devices) [3, 8, 45], and *domain threats and risks* (e.g., modular ERP architecture) [8, 30]. Building on this analysis, the *security design* can be created. To design sufficient and standardized security measures, *security standards* (e.g., ISO/IEC 27000, ISO/IEC 15408) [3, 30, 31, 45] and *data protection* (e.g., privacy regulations) [3, 8, 30] should be considered. The latter mainly refers to means to ensure the privacy of (sensitive) data processed by the ERP product line (e.g., customer or supplier data) [8, 10]. Besides, as *business processes* can be implemented as reusable features [3] that can potentially be misused [31], these should be designed to meet concrete *security requirements* [44]. Note that business processes can be used for security re-engineering, that is analyzing such processes of existing ERP systems to identify and address threats and risks. The collected information should be stored and maintained in a business-process repository connected to a security-requirements repository. Such a setup significantly increases the

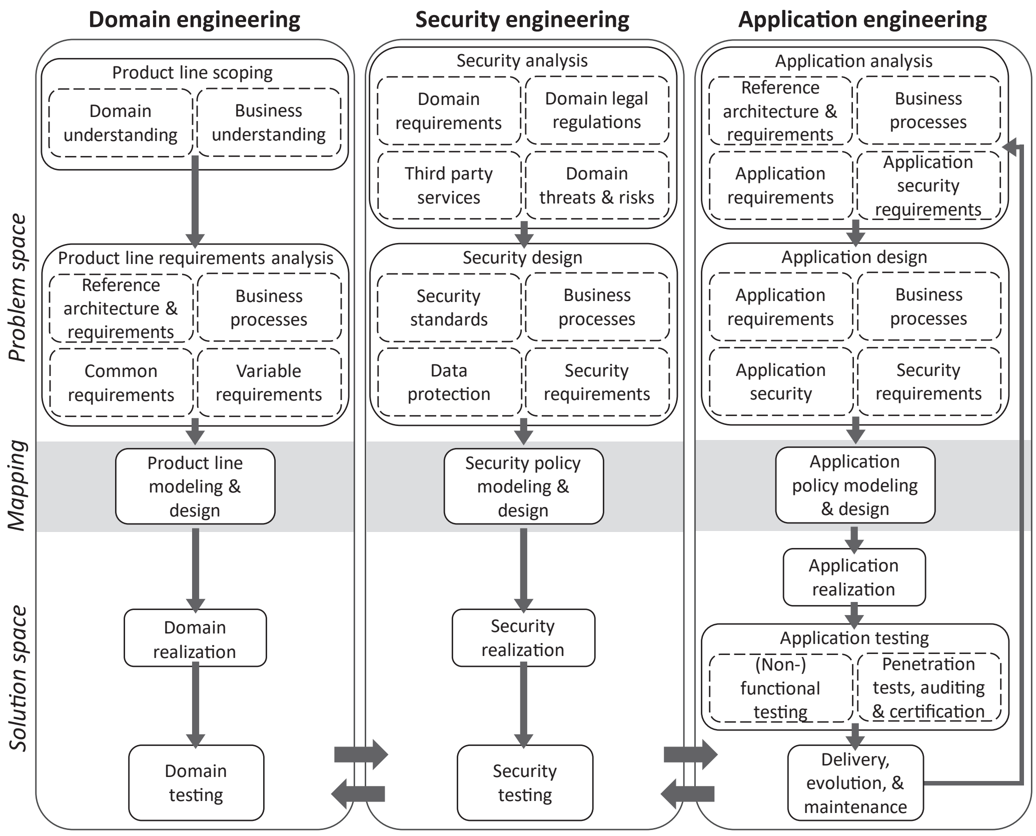

Fig. 2. PLE framework for secure ERP systems, including three perspectives (bold font on top), processes (rounded boxes with solid lines), and activities (rounded boxes with dashed lines) organized according to problem space, mapping, and solution space.

traceability of security measures, of both abstract (e.g., authentication, authorization) and concrete (e.g., fingerprints) ones—particularly in the context of change management [8, 44].

Mapping. Again, to map problem and solution space, our framework involves the process of *security policy modeling and design*. At this points, a company has to create documents for each business process to define how the respective security is managed (e.g., security measures for supply chain management). These documents should include rules for protecting the ERP system, as well as how security breaches and attacks are handled [3, 30, 43].

Solution Space. The last two processes for security engineering are in the solution space: *security realization* and *security testing*. Both processes are equivalent to those in the solution space of the domain engineering. They differ only in their focus on implementing, validating, and verifying reusable security features, services, and policies [30].

4.3. Application Engineering

Problem Space. In the application engineering, the requirements from the domain and security engineering are integrated into a secure ERP variant of the product line. Deriving such a variant starts with an *application analysis* [30, 31], comprising the identification, analysis, and documentation of the ERP *reference architecture and its requirements* (e.g., product layers) [17], *business processes* (e.g., selection of suitable business processes) [45], *application requirements* (e.g., features), and *application security requirements* (i.e., variant-specific requirements) [3, 31]. The following *application design* process refers to all activities for deriving the application architecture [10, 30]. This includes the integration of all *application requirements*, *security requirements* (e.g., considering standards), the overall *application security* (i.e., application-specific security features), and *business processes* (i.e., adaptation to use case) [3, 30, 44].

Mapping. To map problem and solution space, an *application policy modeling and design* to integrate features and security is performed. This process considers the security-engineering policies as well as security and application requirements. Also, configurability-related information and regulations must be covered (e.g., feature updates).

Solution Space. Similar to the other two perspectives, the solution space involves the *application realization* (i.e., deriving a variant from the product line) and *application testing* [3, 30]. However, the testing process involves two activities: *functional and non-functional testing* (i.e., evaluating functionality and quality, respectively) [10] as well as security-related *penetration testing, auditing, and certification* [8]. This way, a variant is evaluated regarding whether it meets the defined domain and product requirements as well as whether it complies with security standards and measures. The last process is the final variant's *delivery, evolution, and maintenance* after its successful testing [3, 10, 12, 43]. In case the product did not meet the specifications, it needs to be iteratively revised and retested [3].

5. Discussion

In this section, we discuss theoretical and practical implications as well as threats to validity.

5.1. Theoretical and Practical Implications

We proposed a framework that guides organizations in considering security while engineering a configurable ERP system. Applying our framework helps include security requirements early into the development process. By relying on PLE, requirements changes in the domain, in the security environment, or at the customer can be dynamically addressed by design. Similar advantages arise regarding the ERP system's evolution. From a company perspective, these advantages offer opportunities in terms of reusability and efficient maintenance, ideally leading to time and cost savings. From the developer perspective, we expect similar advantages in addition to facilitating the introduction of ERP-system families. These families consist of variants sharing business-process features (e.g., sales), including necessary security features based on industry- and customer-specific demands. To increase the efficiency of the engineering process, we recommend to use interconnected business-process and (security) requirements repositories, to document and trace best practices, to design an overall business case, and to define requirements' relationships.

We are aware that comprehensive documentation and detailed policies are often neglected, both in theory and in practice. Accordingly, our framework essentially represents an idealized PLE setting for engineering secure ERP systems. Note that our framework does not include privacy regulations (e.g., GDPR) and their impact on the individual processes and activities. Furthermore, we did not map security standards to the processes, in particular security engineering. So, from the research perspective, we aim to revise and extend our framework by investigating the ERP reference architecture, its influences, and emerging security concerns that need to be addressed in the engineering processes in more detail. However, we argue that our framework already offers value regarding a high-level classification and understanding of how security can be considered in a reusable way while developing ERP systems.

5.2. Threats to Validity

Internal Validity. We cannot rule out that we misinterpreted some terminology the authors of the papers we investigated used (e.g., data protection vs. data privacy). A similar threat arises with respect to the framework perspectives, which were conceptualized at a high-level, but partly explained on a more detailed perspective. Some authors explained their framework's characteristics in great detail, while others only generally mentioned them. In all of such cases, we tried to avoid misinterpretations by using card sorting, comparing the interpretations among all authors, and consulting additional literature on ERP systems or PLE. While we aimed to mitigate this threat via such means, we may still have wrongly classified terms. None of the analyzed frameworks involves all perspectives, processes, or activities. While this fact emphasizes the novelty and value of our work, we may have errors in our synthesis process.
External Validity. Regarding the number of analyzed frameworks, we are aware that more papers could have strengthened the validity of our results. In this context, due to the overall search strategy (i.e., search string and inclusion criteria), we potentially missed papers that are related to sub-topics that could provide more details for our framework (e.g., risk assessment in ERP systems). Finally, we note that our framework has not been evaluated in practice, yet. However, a comprehensive evaluation is planned for future research.

We aim to mitigate these threats by relying on a systematic methodology combining two construction strategies and involving experienced researchers and practitioners in the identification, synthesis, and discussion process. Furthermore, our framework is based on an analysis of papers fetched from three established databases that list peer-reviewed papers. We argue that the qualitative analysis of pre-existing frameworks is useful, even if their amount seems low, since they are build on in-depth domain-specific knowledge and are widely accepted in the research community. So, our framework offers highly valuable insights on the intersection of PLE, ERP systems, and security concerns.

6. Related Work

Different frameworks and process models have been proposed for PLE [5, 22]. Recent works involve more detailed processes than our framework, and can serve as more concrete guidelines on how to engineer configurable systems [22]. However, established frameworks for PLE do not include any perspective on security as a highly relevant concern in software engineering. Other frameworks, which we found during our literature review, aim to combine certain perspectives of PLE and security. For example, Kim et al. [17] have proposed the DRAMA framework for modeling PLE-oriented domain architectures, mentioning security as one quality attribute. Mellado et al. [30, 31, 32] have focused on security requirements engineering frameworks for PLE oriented towards security standards. Alam et al. [3] introduced a framework for developing secure configurable systems, involving security and configurability in the context of reusing artifacts. Varela-Vaca et al. [43] have proposed the CyberSPL framework, focusing on PLE-oriented security-policy compliance of system configurations. Regarding ERP systems, we found several frameworks with connections to security. For instance, Bibi and Saleem [8] have suggested an ERP framework in the context of security risk prevention. Goel et al. [12] have introduced an ERP system engineering framework for technical educational institutions, describing security as one essential feature. Wang et al. [44] have developed a security-management framework for ERP systems based on authentication workflows. Binu and Meenakumari [9], Zhong and Rohde [45], and Chandrakumar and Parthasarathy [10] describe the development of evaluation frameworks for cloud systems. Regarding ERP and configurable systems, ERP systems typically serve as a case study, but are not investigated specifically. For example, Ali et al. [4] have presented a product line for modeling and building ERP systems.

Overall, the related work has integrated security with PLE or ERP systems to some extent, and is partly related to our framework. In contrast to our work, none of the existing frameworks is oriented towards the overall engineering process, taking into account the three relevant perspectives with their processes, activities, and dependencies in a uniform way. Moreover, existing research has not focused on using PLE for developing secure ERP systems, which we argue poses novel challenges due to the intersection of configuration options and critical data. Our framework provides a complement allowing organizations to map security aspects and processes for ERP systems to PLE.

7. Conclusion

In this paper, we introduced a PLE framework for secure ERP systems oriented towards three relevant engineering perspectives and involves 16 processes as well as 24 activities. We emphasize the novelty and value of our framework for researchers and practitioners since it provides a cohesive overview on how to engineer secure and configurable ERP systems. The framework provides a supportive means for improving customer-oriented maintainability and traceability of features by relying on reusable artifacts that include security. We see great benefits for our framework particularly in the context of change management and associated evolutionary changes that influence a system's security feature configurations (e.g., system patches due to system vulnerabilities). In future research, we will refine and evaluate our framework, for example, by incorporating more fine-grained development activities and security standards.

References

[1] Abal, I., Melo, J., Stănciulescu, Ş., Brabrand, C., Ribeiro, M., Wąsowski, A., 2018. Variability bugs in highly configurable systems: A qualitative analysis. Transactions on Software Engineering and Methodology 26.
[2] Acher, M., Bécan, G., Combemale, B., Baudry, B., Jézéquel, J.M., 2015. Product lines can jeopardize their trade secrets, in: ESEC/FSE, ACM.
[3] Alam, M.M., Khan, A.I., Zafar, A., 2017. A secure framework for software product line development. Jorunal of Computer Applications 975.
[4] Ali, M., Nasr, E.S., Geith, M.H., 2016. A requirements elicitation approach for cloud based software product line ERPs, in: FAMECSE, ACM.
[5] Apel, S., Batory, D., Kästner, C., Saake, G., 2013. Feature-oriented software product lines. Springer.
[6] Asif, A., AlFrraj, D., Alshamari, M.A., 2022. A comprehensive approach of exploring usability problems in enterprise resource planning systems. Applied Sciences 12.
[7] Bakry, A.H., Bakry, S.H., 2005. Enterprise resource planning: A review and a STOPE view. Journal of Network Management 15.

[8] Bibi, S., Saleem, N., 2009. Proposed security framework for ERP systems. Journal of Independent Studies and Research 7.

[9] Binu, S., Meenakumari, J., 2012. A security framework for an enterprise system on cloud. Journal of Computer Science and Engineering 3.

[10] Chandrakumar, T., Parthasarathy, S., 2014. A framework for evaluating cloud enterprise resource planning (ERP) systems. Continued Rise of the Cloud: Advances and Trends in Cloud Computing 1.

[11] Chang, B.R., Tsai, H.F.F., Tsai, Y.C., Chang, Y.S., 2014. Applying authentication and network security to in-cloud enterprise resource planning system. Vietnam Journal of Science, Technology and Engineering 1.

[12] Goel, S., Kiran, R., Garg, D., 2011. A framework for efficient ERP implementation in technical educational institutions. African Journal of Business Management 5.

[13] Jamshidi, P., Velez, M., Kästner, C., Siegmund, N., Kawthekar, P., 2017. Transfer learning for improving model predictions in highly configurable software, in: SEAMS, ACM.

[14] Katuu, S., 2021. Managing records in enterprise resource planning systems, in: Big Data, IEEE.

[15] Kenner, A., May, R., Krüger, J., Saake, G., Leich, T., 2021. Safety, security, and configurable software systems: A systematic mapping study, in: SPLC, ACM.

[16] Khaleel, Y.K., Alkhaldi, A.N., 2017. ERP model for small and medium sized manufacturing firms based on UML. International Business Journal 9.

[17] Kim, J., Park, S., Sugumaran, V., 2008. DRAMA: A framework for domain requirements analysis and modeling architectures in software product lines. Journal of Systems and Software 81.

[18] Kitchenham, B.A., Budgen, D., Brereton, O.P., 2015. Evidence-based software engineering and systematic reviews. CRC Press.

[19] Krueger, C.W., 2002. Easing the transition to software mass customization, in: PFE, Springer.

[20] Krüger, J., 2021. Understanding the re-Engineering of variant-rich systems: An empirical work on economics, knowledge, traceability, and practices. Ph.D. thesis. Otto-von-Guericke University Magdeburg.

[21] Krüger, J., Berger, T., 2020. An empirical analysis of the costs of clone- and platform-oriented software reuse, in: ESEC/FSE, ACM.

[22] Krüger, J., Mahmood, W., Berger, T., 2020. Promote-pl: A round-trip engineering process model for adopting and evolving product lines, in: SPLC, ACM.

[23] Langenwalter, G.A., 2020. Enterprise resources planning and beyond: integrating your entire organization. CRC Press.

[24] van der Linden, F.J., Schmid, K., Rommes, E., 2007. Software product lines in action. Springer.

[25] Maheshwari, S., Sharma, C., 2014. Ten security practices to a formidable ERP system, in: ICSSS, IEEE.

[26] May, R., 2022. Security and configurable storage systems in industry 4.0 environments: A systematic literature study, in: OCP.

[27] May, R., Biermann, C., Krüger, J., Saake, G., Leich, T., 2022. A systematic mapping study of security concepts for configurable data storages, in: SPLC, ACM.

[28] May, R., Gautam, J., Sharma, C., Biermann, C., Leich, T., 2023. A systematic mapping study on security in configurable safety-critical systems based on product-line concepts, in: ICSOFT, SciTePress.

[29] Mazo, R., Assar, S., Salinesi, C., Hassen, N.B., 2014. Using software product line to improve ERP engineering: literature review and analysis. Latin-American Journal of Computing 1.

[30] Mellado, D., Fernández-Medina, E., Piattini, M., 2008. Towards security requirements management for software product lines: A security domain requirements engineering process. Computer Standards & Interfaces 30.

[31] Mellado, D., Fernández-Medina, E., Piattini, M., 2010. Security requirements engineering framework for software product lines. Information and Software Technology 52.

[32] Mellado, D., Mouratidis, H., Fernández-Medina, E., 2014. Secure tropos framework for software product lines requirements engineering. Computer Standards & Interfaces 36.

[33] Nadi, S., Berger, T., Kästner, C., Czarnecki, K., 2014. Mining configuration constraints: Static analyses and empirical results, in: ICSE, IEEE.

[34] Nešić, D., Krüger, J., Stănciulescu, S., Berger, T., 2019. Principles of feature modeling, in: ESEC/FSE, ACM.

[35] Nickerson, R.C., Varshney, U., Muntermann, J., 2013. A method for taxonomy development and its application in information systems. European Journal of Information Systems 22.

[36] Rodlauer, J., Junghans, S., Trommer, M., Leonhardt, S., 2022. Integration of the resource of electric energy into enterprise-resource-planning for the compliance of EU policies, in: UCAml, Springer.

[37] Saeed, I., Juell-Skielse, G., Uppström, E., 2012. Cloud enterprise resource planning adoption: Motives & barriers. Advances in Enterprise Information Systems 429.

[38] Salih, S., Hamdan, M., Abdelmaboud, A., Abdelaziz, A., Abdelsalam, S., Althobaiti, M.M., et al., 2021. Prioritising organisational factors impacting cloud ERP adoption and the critical issues related to security, usability, and vendors: A systematic literature review. Sensors 21.

[39] Shehab, E.M., Sharp, M.W., Supramaniam, L., Spedding, T.A., 2004. Enterprise resource planning: An integrative review. Business Process Management Journal 1.

[40] Singh, K., Best, P.J., 2015. Design and implementation of continuous monitoring and auditing in SAP ERP. Journal of Auditing. 19.

[41] Tarhini, A., Ammar, H., Tarhini, T., Masa'deh, R., 2015. Analysis of the critical success factors for enterprise resource planning implementation from stakeholders' perspective: A systematic review. International Business Research 8.

[42] Tavana, M., Hajipour, V., Oveisi, S., 2020. IoT-based enterprise resource planning: Challenges, open issues, applications, architecture, and future research directions. Internet of Things Journal 11.

[43] Varela-Vaca, Á.J., M. Gasca, R., Ceballos, R., Gómez-López, M.T., Bernáldez Torres, P., 2019. CyberSPL: A framework for the verification of cybersecurity policy compliance of system configurations using software product lines. Applied Sciences 9.

[44] Wang, F., Ge, B., Zhang, L., Chen, Y., et al., 2013. A system framework of security management in enterprise systems. Systems Research and Behavioral Science 30.

[45] Zhong, F., Rohde, M.E., 2014. Cloud computing and ERP: A framework of promises and challenges, in: ACIS, AAIS.