Generation of IT Project Documentation Elements from a Model Transformation Chain

Oksana Nikiforova[1][a], Megija Krista Miļūne[1][b], Kristaps Babris[1][c] and Oscar Pastor[2][d]
[1]*Riga Technical University, Riga, Latvia*
[2]*Universitat Politècnica de València, Valencia, Spain*

Keywords: IT Project Artefacts (Artifact), Model Transformation Chain, IT Project Management, IT Project Initiating, IT Project Planning.

Abstract: Documentation plays a crucial role in IT project management, particularly in the early stages, where it helps define requirements, establish goals, and mitigate risks. Ensuring documentation quality and consistency remains a challenge, necessitating adherence to established standards such as IEEE 830 and ISO/IEC 12207. In our previous research, we proposed a model transformation chain to facilitate the generation of IT project artefacts, which are offered as a solution in last 5 years scientific papers. This paper expands on that work by examining how documentation elements can be systematically extracted and structured from the transformation chain more detailed representation. We discuss the elements of documentation that can be derived and mapping components for extracting relevant information from the artefacts already developed in the project. Our findings highlight the potential to improve documentation quality and reduce manual effort. A case study demonstrates the practical application of our framework to a small-scale IT project documentation.

1 INTRODUCTION

Effective documentation is a critical aspect of IT project management, serving as a foundation for communication, decision-making, and knowledge retention throughout the project lifecycle (Jarzębowicz and Weichbroth, 2021). In the early stages of an IT project, well-structured documentation is particularly essential, as it guides stakeholders in defining requirements, establishing goals, and mitigating risks before implementation begins. However, obtaining consistent and high-quality documentation remains a challenge, particularly when dealing with complex and evolving project artefacts.

Furthermore, IT project documentation must adhere to established standards and best practices to ensure consistency, completeness, and usability across different stakeholders and project phases. Compliance with standards such as IEEE 830 for software requirements specifications (IEEE, 1994) or ISO/IEC 12207 for software lifecycle processes (ISO, 2008) facilitates better communication, reduces ambiguities, and enhances the overall reliability of documentation. By aligning documentation with recognized frameworks, organizations can improve collaboration, ensure regulatory compliance, and streamline project execution.

In our previous research (Nikiforova et al., 2025b) we introduced a model transformation chain designed to facilitate the generation of IT project artefacts. Solutions published in 133 studies offering IT project elements obtaining with principles of model-driven engineering, machine learning and generative artificial intelligence (AI), as well as manual practices, were mapped into artefacts used in the initial stages of IT project identified in (Nikiforova et al., 2025a). This transformation chain provides a structured framework, where relevant project elements are obtained ones from another ones,

[a] https://orcid.org/0000-0001-7983-3088
[b] https://orcid.org/0009-0002-2417-4518
[c] https://orcid.org/0000-0003-3855-6963
[d] https://orcid.org/0000-0002-1320-8471

Nikiforova, O., Miļūne, M. K., Babris, K. and Pastor, O.
Generation of IT Project Documentation Elements from a Model Transformation Chain.
DOI: 10.5220/0013568300003964
In *Proceedings of the 20th International Conference on Software Technologies (ICSOFT 2025)*, pages 336-345
ISBN: 978-989-758-757-3; ISSN: 2184-2833

ensuring alignment with project objectives and stakeholder expectations. By leveraging this transformation chain, we aim to improve the efficiency and accuracy in the initial stages of IT projects.

This paper follows our previous work by focusing on the extraction and structuring of agile IT project elements (Agile manifesto 2001) as mapping of them into the created model transformation chain. We explore the elements of documentation that can be derived, the methodologies employed to extract relevant information, and the benefits of integrating this approach into IT project management practices. Through this study, we aim to provide a systematic framework that enhances documentation development reducing manual effort and inconsistencies.

The remainder of this paper is organized as follows: Section 2 provides an overview of related work in IT project documentation and model-driven approaches. Section 3 briefly describes the Agile IT project life cycle and IT Project Management Plan structure, which is used for mapping in the next section. Section 4 specify the fragments of the model-transformation chain in details for extracting documentation elements in IT Project Management Plan. Section 5 presents a case study demonstrating the practical application of our approach. Finally, Section 6 discusses the findings, implications, and future research directions.

2 RELATED WORK

Project documentation of the IT project area is crucial to the success of a project. This requires a systematic plan to extract the necessary elements from the initial artefacts. Many methodologies have been proposed to help obtain documentation in a structured way in IT projects.

Retrieving and structuring IT project documentation can be accomplished through different levels of automation. Manual processing relies entirely on human effort for gathering, structuring, and maintaining documentation. Semi-automatic combines manual effort with automated tools to facilitate documentation generation. Fully automated systems utilize AI, ML, and Natural Language Processing (NLP) to extract, classify, and generate documentation artefacts without manual intervention (Prasetyo et al., 2025).

One of the approaches involves the use of knowledge extraction (Schlutter and Vogelsang, 2020) and NLP methodologies (Ferrari et al., 2021), which allow for automated organisation of documentation. Text mining techniques (Talib et al., 2016) help to verify requirements documents, identify primary entities and retrieve relationships within project components. Ontology-based structuring (Bencharqui, 2022) ensures consistency, while Named Entity Recognition (Das et al., 2023) enhances traceability by identifying project-specific terms.

AI (Al-Arafat et al., 2025) and ML models (Thota et al., 2024) have been introduced to improve documentation accuracy by learning patterns in requirement specifications, suggesting relevant content and to classify and categorize project documentation. These models, leveraging deep learning and neural networks (Dehaerne et al., 2022), help in reducing inconsistencies and enhancing automation in documentation generation.

However, challenges such as contextual ambiguity (AI and NLP-based techniques struggle to fully understand context-dependent requirements), domain-specific variations, and the need for large, labelled datasets make it difficult to achieve fully automated and error-free documentation extraction.

Model-driven development (MDD) offers a step-by-step approach (Pastor and Molina, 2007) to generating IT project documentation introducing a multi-layered transformation process (Pastor et al., 2021). Requirements are first captured using structured representations (Alkhatib, 2024) such as Unified Modeling Language (UML) or Business Process Model and Notation (BPMN) which can be retrieved from Natural Language (Sholiq, 2022), (Meng and Ban, 2024), (Nikiforova & Pavlova, 2009). These are transformed into PIM and then refined into PSM, ensuring alignment with system implementation needs (Nikiforova et al., 2009). Automated tools such as Papyrus, MagicDraw, and Enterprise Architect facilitate the extraction of textual documentation from models, reducing manual effort and improving accuracy. However, different documentation tools and methodologies often lack seamless integration.

Ensuring compliance with established standards (for example, ISO/IEC 12207 or IEEE 828) remains challenging and requires constant updates and alignment with evolving industry practices. Model-driven development relies on the creation of conceptual models at various levels of abstraction and the use of transformation models to systematically transition from higher-level to lower-level representations. This approach follows the Model-driven Architecture (MDA) pipeline, which begins with a Computation-Independent Model (CIM). The

CIM is then transformed into Platform-Independent Model (PIM), which, through its corresponding Platform-Specific Model (PSM), ultimately generates the final application code. This structured process, which guides model-driven software development, is referred to as a model transformation chain. Transformation chains used in artefact generation (Pastor et al., 2022) must maintain consistency across different project phases, necessitating sophisticated traceability mechanisms and integrating multiple methodologies, such as MDD (Noël et al., 2022) and NLP, what remains complex due to differences in data formats and process workflows. Moreover, large projects generate vast amounts of documentation, making automated extraction complex while inconsistencies in input data can lead to documentation errors.

3 AGILE IT PROJECT DOCUMENTATION

In the early stages of an agile IT project, documentation focuses on establishing a shared understanding of the product vision and setting the stage for iterative development (PMI, 2021). Unlike the rigid, upfront planning of traditional methodologies, agile emphasizes "just enough" documentation to enable rapid adaptation and collaboration (Behutiye et al., 2022). The primary goal is to capture the essence of the product and its intended value, rather than producing exhaustive documentation (Pasuksmit et al., 2021). This approach promotes flexibility and responsiveness to changing requirements.

Key agile artefacts produced during these early stages include the product vision and roadmap. The product vision provides a high-level overview of the product's goals and target audience, while the roadmap outlines the planned releases and key features over time (Chantaravisutlert, 2022). Backlog management, involving the creation and prioritization of user stories, is also crucial (PMI, 2021). These artefacts are typically maintained in collaborative tools like Jira (Atlassian), allowing for continuous refinement and adaptation. However, challenges arise in ensuring these artefacts are easily retrievable and consistently updated, especially in distributed teams.

It's important to note that the specific documents required will vary depending on the project's size, complexity, and the methodology used (Mesjasz et al., 2022).

Artefacts described in agile documentation (Figure 1) standards prioritize clarity, conciseness, and collaboration. User stories, for example, are written in a simple, user-centric format, providing acceptance criteria that must be met for a user story to be considered complete (Kuhail and Lauesen, 2022). To manage scope, the product backlog is segmented into smaller, manageable pieces, allowing for iterative development and frequent feedback. Prioritization techniques help to focus on the most valuable features.

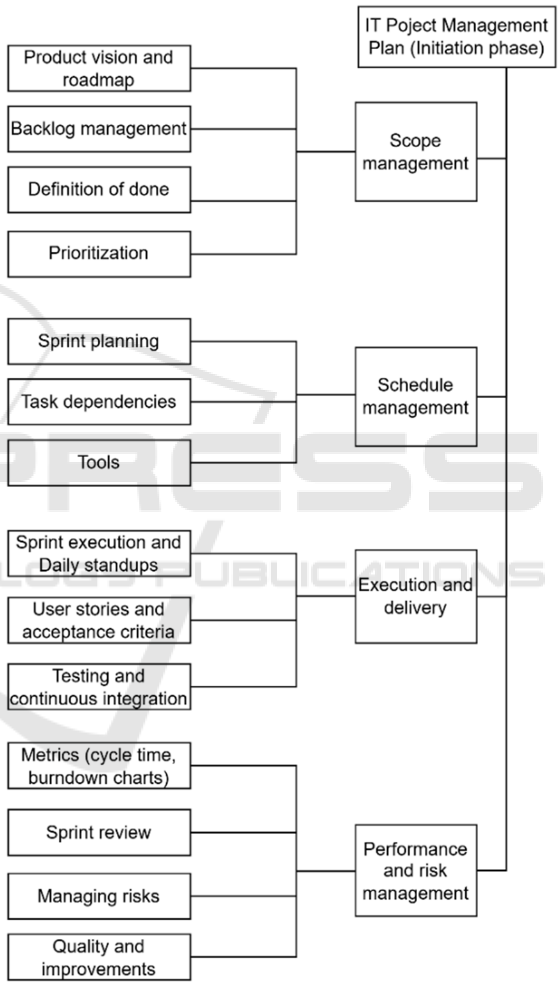


Figure 1: Activities performed under the IT project management plan.

Schedule management (PMI, 2021) in agile projects revolves around sprint planning. Task dependencies are visualized through sprint backlogs and Kanban boards (Agile Alliance), while tools like Jira and Azure DevOps (Azure) facilitate task tracking and progress monitoring. Sprint planning sessions are documented through sprint backlogs, which detail the tasks to be completed during the sprint. These documents are living artefacts, updated daily to reflect progress and changes (Marnada et al., 2022). The challenge here is keeping these documents up to date and accessible to all team members, especially in fast-paced environments.

Execution and delivery in agile are characterized by sprint execution and daily stand-ups. User stories and acceptance criteria are the primary documentation for development tasks, ensuring that the team understands what needs to be delivered (Baïna et al., 2020). Testing and continuous integration are integral parts of the development process, with test cases and automated scripts documenting the testing efforts. These documents are generally kept within the CI/CD pipeline and are referenced when bugs arise. The challenge, however, is to ensure that these documents are easily accessible and understandable to all team members, not just the developers.

Performance and risk management in agile rely on metrics like cycle time and burndown charts, which provide insights into the team's progress and performance. Sprint reviews are used to gather feedback from stakeholders and demonstrate the completed work. Managing risks, quality, and improvements is an ongoing process, with retrospectives documenting lessons learned and action items for improvement (Garcia et al., 2022). The challenge lies in maintaining a balance between capturing necessary information and avoiding excessive documentation that hinders agility. These documentation pieces can be broken down into small, digestible chunks, like individual user stories, sprint reviews, or risk assessments, which are then stored in a central repository, allowing for easier retrieval and updates.

4 MODEL TRANSFORMATION CHAIN FOR IT PROJECT INITIATION ARTEFACTS

In previous research (Nikiforova et al., 2025b) we gathered large model transformation chain which can be applied for obtaining IT project documentation artefacts at the initial stages of the project before software implementation. In this article, we take and examine in more detail a fragment of this large chain that related to project documentation artefacts according to the IT Project Management Plan.

This section is organized as follows. Section 4.1. shows transformation chain fragment for Scope Management, describing in details methods used for artefacts obtaining. The rest of subsections demonstrates only transformation chain types. By dotted arrows in the transformation chains figures authors highlight manual transformations and by solid arrow the transformations with ability to automate them are shown.

4.1 Transformation Chain for Scope Management

Figure 2 shows all the possible transformations among artefacts of the IT Project Management Plan section Scope Management.

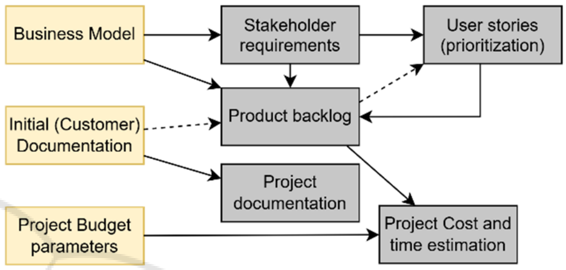


Figure 2: Artefacts transformation chain for Scope Management.

From Business Model artefact it is possible to obtain Stakeholder Requirements by using direct transformation of processes and their performers into actors and use cases of UML use case diagram. Moreover, (Jarzębowicz and Weichbroth, 2021) offers a solution for working with non-functional requirements. As well as Product Backlog can be defined by methods of current state analysis, continuous re-engineering, reverse engineering described in (Doshi and Virparia, 2023). Manually Product Backlog can be refined from information of Initial (Customer) Documentation as it is explained in (Mohammad and Kollamana, 2024).

However, Initial (Customer) Documentation can be used for automatic refinement of Project Documentation as it is explained in (Nagoya, 2021).

The fuzzy model for cost and time optimization described in (Kaushik et al., 2020) utilizes a triangular membership function to automatically derive Project Cost and Time Estimation from Project Budget Parameters.

Thus, the Product Backlog can be automatically defined using the Design Thinking model described in (Alhazmi and Huang, 2020), which incorporates Stakeholder Requirements. User stories (prioritization) can also be obtained from Stakeholder Requirements showing automatic transformations in

(Wagner and Ford, 2020) using suitability of tools or DevOps practices to fulfill the requirements.

Transformation from User Stories (prioritization) to Product Backlog is automatic by using the SCSE framework shown in (Kusdiyanto et al., 2022). As well as using machine learning techniques in (Rodríguez Sánchez, et al., 2023) shows automatic transformations from Product Backlog to Project Cost and Time Estimation.

The final transformation for Scope Management gathers data from the Product Backlog to User Stories (prioritization) when the prioritization is conducted manually through project vision meetings described in (Galván-Cruz, et al., 2021).

4.2 Transformation Chain for Schedule Management

Figure 3 shows all the possible transformations among artefacts of the IT Project Management Plan section Schedule Management.

The transformation from Initial (Customer) documentation to User Story and Sprint Backlog is manual, the same as is the transformation from Product Backlog to User Story and back. Additionally, manual transformation occurs from Process Organization to User Story, from User Story to Project Team and finally from Project Team to Sprint Backlog artefact. The remaining transformations for Schedule Management are possible to obtain with automatization methods described in (Galván-Cruz, et al., 2021), (Sayeb et al., 2024), (Noreika and Gudas, 2021).

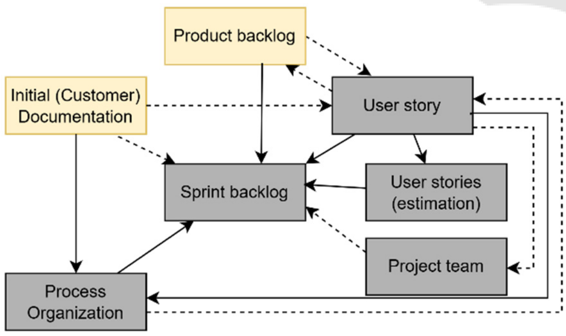


Figure 3: Artefacts transformation chain for Schedule Management.

4.3 Transformation Chain for Execution and Delivery

Figure 4 shows all possible transformations among the artefacts in the Execution and Delivery section of the IT Project Management Plan.

Of the artefacts defined in this section, only the transformation from Initial (Customer) documentation to Process Organization is highlighted as automated. A solution for this automated transformation is offered in (Sayeb et al., 2024). The remaining transformations described in Execution and Delivery section are obtained manually.

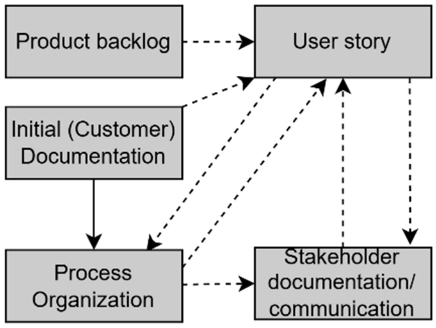


Figure 4: Artefacts transformation chain for Execution and Delivery.

4.4 Transformation Chain for Performance and Risk Management

Figure 5 shows all the possible transformations among the artefacts in the Performance and Risk Management section of the IT Project Management Plan.

The transformation from User Story to Security, as well as from Process Organization and User Stories (estimation) to Risk Management, can be obtained using automatization methods, as discussed in (Herwanto et al., 2024), (Cabrero-Daniel et al., 2024), (Neto et al., 2023). However, remaining transformations in this section can be executed manually.

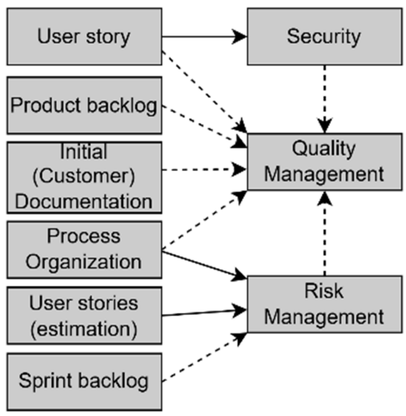


Figure 5: Artefacts transformation chain for Performance and Risk Management.

5 TRANSFORMATION EXAMPLE DEMONSTRATION

For the demonstration of the transformation chain described in the paper, authors selected a fragment of Project Scope definition. Its transformations are shown in Figure 6. For practical example demonstration the problem domain is chosen, where students select thesis topic for their research. Business process model of the problem domain is presented in Figure 7 (a).

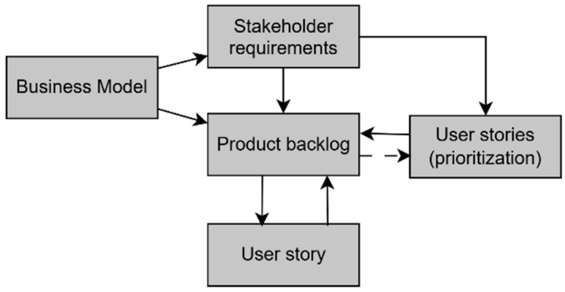
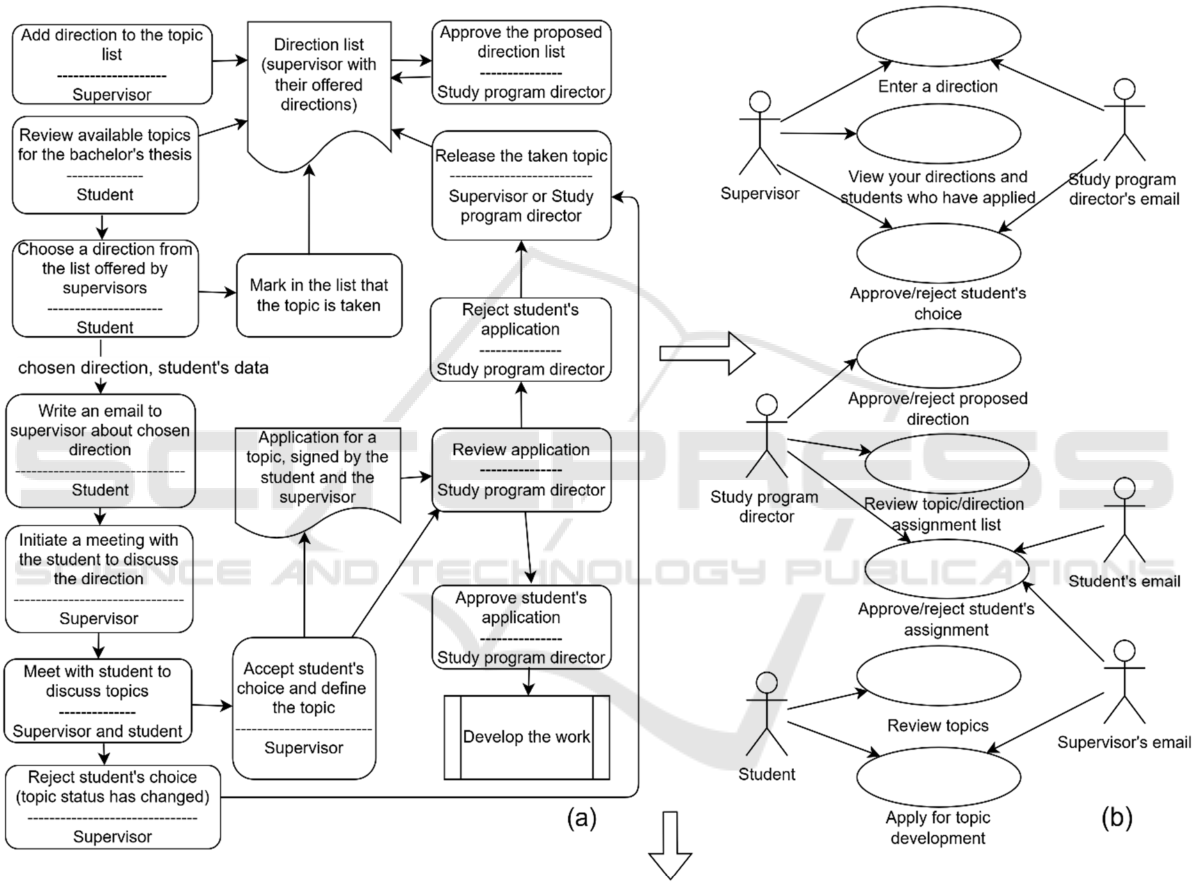


Figure 6: Project Scope definition transformation chain.



Priorities	As	<<role>>	I want	<<use case>>	<<context>>
1	As	a student	I want	to review the list of directions	to choose one for development.
3	As	a student	I want	to group the directions by supervisor, keywords or complexity level	to structure them, make them more organized, and simplify the selection process.
2	As	a student	I want	to review detailed information of directions	to get a more detailed understanding of the specific direction and determine whether it is interesting.
2	As	a student	I want	to review detailed information from the supervisor	to gain a more detailed understanding whether it is suitable.
1	As	a student	I want	to apply for the selected direction	to initiate work/ take the first step for myself
2	As	a student	I want	to withdraw from the selected direction	to choose other direction.

(c)

Figure 7: Business process model of the problem domain (a), the corresponding UML use case diagram (b) and the obtained product backlog with prioritized user stories (c).

As far as for transformations presented in Figure 6, the Business Model is used as the initial information into transformation chain. From this model, stakeholder requirements are obtained in the form of the UML use case diagram. This transformation can be implemented automatically (highlighted as solid row between the artefacts in Figure 6), e.g., in correspondence with the approach offered in (Nikiforova et al., 2017). The requirements expressed in the form of system use cases are transformed into user stories, which become part of the product backlog. These user stories are then prioritized in the correspondence with methods described in Section 4.1.

As far as for practical example, the processes performed by actors in the business model in Figure 7 (a) are transformed into particular use cases associated with the corresponding actors. A fragment of the UML use case diagram, obtained from the presented business process diagram, is shown in Figure 7 (b). The definition of Product Backlog, based on the business model and stakeholders' requirements, is suggested as automation. In this automation, actors and use cases are transformed into the form of user stories. However, this automation should be combined by manual input from stakeholders, who will then prioritize the user stories for implementation. The project user stories are documented and listed in the table in Figure 7 (c). User stories are translated from the provided use cases and integrated into the project documentation.

Once the product backlog is created, it can be imported into project management tools, such as Jira for further project planning and development. This integration can help to minimize the risk of deviations from project requirements during implementation.

6 DISCUSSION AND CONCLUSIONS

Obtaining consistent and well-structured IT project documentation remains a persistent challenge in both academic and industrial contexts. Project managers and development teams must often deal with a large volume of evolving artefacts, which are produced and modified throughout the software development lifecycle. These artefacts are not always aligned with one another, and the absence of a structured documentation framework can lead to miscommunication, duplicated effort, and an increased risk of project failure. This issue is particularly pronounced in agile or hybrid project environments, where documentation is often deprioritized in favour of rapid delivery cycles. Nevertheless, clear, consistent, and standards-compliant documentation remains essential for stakeholder communication, risk mitigation, traceability, and long-term maintainability.

This paper addresses the need for a more systematic and automated approach to documentation generation by introducing a model transformation chain aimed at structuring documentation elements in alignment with the IT Project Management Plan. Our approach leverages existing artefacts – such as business process models, user stories, or requirement specifications and applies transformation logic to extract and organize relevant content. The proposed method helps reduce redundancy, improve traceability, and align outputs with established standards, such as IEEE 830 and ISO/IEC 12207.

While our findings demonstrate that many transformations between IT project artefacts can be automated, certain transformations – particularly those that involve semantic interpretation or abstract reasoning – still require manual intervention. This gap in automation is especially evident in the early phases of IT projects, where ambiguity is highest, and artefacts are often incomplete or inconsistently defined.

The application of AI in early-phase documentation tasks, such as requirements gathering or process modelling, is constrained by the lack of true semantic understanding. Most contemporary AI tools, especially those based on Large Language Models, operate by identifying statistical correlations in textual data. While these models can produce syntactically correct content, their outputs often lack domain-specific accuracy, contextual relevance, and consistency with other artefacts. This becomes particularly problematic when dealing with structured logic or business rules embedded in models such as BPMN or UML diagrams. As a result, AI-generated documentation at this stage may resemble speculative or "fantasy" content rather than actionable, standards-compliant deliverables.

Through our research, we identified transformation chains for four critical sections of the IT Project Management Plan and for each section, we defined mappings between commonly used artefacts and corresponding documentation elements. For instance, in our case study, user stories were successfully derived from business process models to define the project scope. These mappings serve as the foundation for a documentation framework that supports partial automation, reduces manual effort, and improves consistency.

Our findings suggest that IT project artefacts, when properly structured and semantically enriched, can serve as valuable primary sources for documentation generation. We propose integrating documentation as a parallel process embedded within the model-driven development workflow. The transformation chains we propose allow for continuous documentation that evolves alongside the project and remains aligned with its progress.

The proposed approach also facilitates better collaboration between technical and non-technical stakeholders. By translating complex models into human-readable documentation, it becomes easier to involve business users, analysts, and decision-makers in the development process. This not only improves project transparency but also enhances the likelihood of delivering solutions that meet real business needs.

Based on the research presented in this paper, we draw the following conclusions:

- IT project artefacts can be effectively used as primary inputs for structured documentation, ensuring consistency and completeness across various project phases.
- The application of model transformation chains enhances the automation of documentation processes, thereby reducing manual effort, human error, and time spent on repetitive tasks.
- Documentation generation can be systematically updated and maintained by integrating transformation logic directly into the project development lifecycle.

Future work will focus on several key areas. First, we plan to refine and expand the transformation chains to cover additional components of IT project documentation, including stakeholder analysis, quality management, and change control. Second, we will work on defining a more detailed meta-model for IT artefacts to support more accurate mappings and transformations. Third, we aim to explore the integration of domain-specific ontologies and semantic technologies to enhance the interpretability of artefacts by AI systems. Finally, we will continue validating our framework through empirical case studies in both academic and industrial settings, with the goal of contributing to the development of faster, higher-quality, and more maintainable IT project documentation.

ACKNOWLEDGEMENTS

This research has been supported by Research and Development grant No RTU-PA-2024/1-0015 under the EU Recovery and Resilience Facility funded project No. 5.2.1.1.i.0/2/24/I/CFLA/003 “Implementation of consolidation and management changes at Riga Technical University, Liepaja University, Rezekne Academy of Technology, Latvian Maritime Academy and Liepaja Maritime College for the progress towards excellence in higher education, science, and innovation”.

REFERENCES

Agile Alliance, "Kanban," Agile Alliance. [Online]. Available: https://www.agilealliance.org/glossary/kanban/. [Accessed: Apr. 9, 2025]

Agile Manifesto (2001) https://agilemanifesto.org/

Al-Arafat, M., Kabir, M. E., Morshed, A., Islam, M. M. (2025). Artificial Intelligence in Project Management: Balancing Automation and Human Judgment. Frontiers in Applied Engineering and Technology, 2(01), 18–29. DOI: 10.70937/faet.v2i01.47

Alhazmi, A., & Huang, S. (2020). Integrating Design Thinking into Scrum Framework in the Context of Requirements Engineering Management. 3rd International Conference on Computer Science and Software Engineering, 33–45. DOI: 10.1145/3403746.3403902

Alkhatib, G. (2024). Structured Methodologies vs UML Artifacts Revisited: A perspective from a developing country, Procedia Computer Science, Volume 239, Pages 2090-2097, ISSN 1877-0509, DOI: 10.1016/j.procs.2024.06.396

Atlassian, "Jira | Issue & Project Tracking Software," Atlassian. [Online]. Available: https://www.atlassian.com/software/jira. [Accessed: Apr. 9, 2025]

Baïna, K., El Hamlaoui, M., & Kabbaj, H. (2020). Business Process Modelling Augmented: Model Driven transformation of User Stories to Processes. 13th International Conference on Intelligent Systems: Theories and Applications, 1–6. DOI: 10.1145/3419604.3419793

Behutiye, W., Rodríguez, P., Oivo, M., Aaramaa, S., Partanen, J., Abhervé, A. (2022). Towards optimal quality requirement documentation in agile software development: A multiple case study, Journal of Systems and Software, Volume 183, 111112, ISSN 0164-1212, DOI: 10.1016/j.jss.2021.111112

Bencharqui, H., Haidrar, S., Anwar, A. (2022). Ontology-based Requirements Specification Process. E3S Web Conf., 351 01045. DOI: 10.1051/e3sconf/202235101045

Cabrero-Daniel, B., Herda, T., Pichler, V., Eder, M. (2024). Exploring Human-AI Collaboration in Agile: Customised LLM Meeting Assistants. In: Šmite, D.,

Guerra, E., Wang, X., Marchesi, M., Gregory, P. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2024. Lecture Notes in Business Information Processing, vol 512. Springer, Cham. DOI: 10.1007/978-3-031-61154-4_11
Chantaravisutlert. C., Ueasangkomsate, P. (2022). Agile Roadmapping: Systematic Literature Review. In Proceedings of the 4th International Conference on Management Science and Industrial Engineering (MSIE '22). Association for Computing Machinery, New York, NY, USA, 112–117. DOI: 10.1145/35357 82.3535798
Das, S., Deb, N., Cortesi, A., Chaki, N. (2023). Zero-shot Learning for Named Entity Recognition in Software Specification Documents, in 2023 IEEE 31st International Requirements Engineering Conference (RE), Hannover, Germany, pp. 100-110, DOI: 10.1109/RE57278.2023.00019
Dehaerne, E., Dey, B., Halder, S., De Gendt, S., Meert, W. (2022). Code Generation Using Machine Learning: A Systematic Review, in IEEE Access, vol. 10, pp. 82434-82455, DOI: 10.1109/ACCESS.2022.3196347
Doshi, M., Virparia, P. (2023). Agile Development Methodology for Software Re-engineering. In: Goar, V., Kuri, M., Kumar, R., Senjyu, T. (eds) Advances in Information Communication Technology and Computing. Lecture Notes in Networks and Systems, vol 628. Springer, Singapore. DOI: 10.1007/978-981-19-9888-1_32
Ferrari, A., Zhao, L., Alhoshan, W. (2021). NLP for Requirements Engineering: Tasks, Techniques, Tools, and Technologies, 2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), Madrid, pp. 322-323, DOI: 10.1109/ICSE-Companion52605.2021.00137
Galván-Cruz, S., Muñoz, M., Mejía, J., Laporte, C.Y., Negrete, M. (2021). Building a Guideline to Reinforce Agile Software Development with the Basic Profile of ISO/IEC 29110 in Very Small Entities. In: Mejia, J., Muñoz, M., Rocha, Á., Quiñonez, Y. (eds) New Perspectives in Software Engineering. CIMPS 2020. Advances in Intelligent Systems and Computing, vol 1297. Springer, Cham. DOI: 10.1007/978-3-030-63329-5_2
Garcia, F., Hauck, J., & Hahn, F. (2022). Managing Risks in Agile Methods: A Systematic Literature Mapping. 394–399. DOI: 10.18293/SEKE2022-123
Herwanto, G. B., Quirchmayr, G., Tjoa, A. M. (2024). Leveraging NLP Techniques for Privacy Requirements Engineering in User Stories, in IEEE Access, vol. 12, pp. 22167-22189, DOI: 10.1109/ACCESS.2024.33645 33
IEEE Recommended Practice for Software Requirements Specifications (1994). in IEEE Std 830-1993, vol., no., pp.1-32, DOI: 10.1109/IEEESTD.1994.121431
ISO, ISO/IEC 12207: Systems and Software Engineering – Software life Cycle Processes (2008). International Organization for Standardization and International Electrotechnical Commission
Jarzębowicz, A., Weichbroth, P. (2021). A Qualitative Study on Non-Functional Requirements in Agile Software Development, IEEE Access, v. 9, 40458-40475, DOI: 10.1109/ACCESS.2021.3064424
Kaushik, A., Tayal, D.K., Yadav, K. (2020). A Fuzzy Approach for Cost and Time Optimization in Agile Software Development. In: Pati, B., Panigrahi, C., Buyya, R., Li, KC. (eds) Advanced Computing and Intelligent Engineering. Advances in Intelligent Systems and Computing, vol 1082. Springer, Singapore. DOI: 10.1007/978-981-15-1081-6_53
Kuhail, M. A., Lauesen, S. (2022). User Story Quality in Practice: A Case Study. Software, 1(3), 223-243. DOI: 10.3390/software1030010
Kusdiyanto, A., Suhardi, Muhamad, W. (2022). Combining Services Computing Systems Engineering Framework and Scrum for Agile Development. 2022 International Conference on Information Technology Systems and Innovation (ICITSI), Bandung, Indonesia, pp. 12-18, DOI: 10.1109/ICITSI56531.2022.9970965
Marnada, P., Raharjo, T., Hardian, B., Prasetyo, A. (2022). Agile project management challenge in handling scope and change: A systematic literature review. Procedia Computer Science, Volume 197, Pages 290-300, ISSN 1877-0509, DOI: 10.1016/j.procs.2021.12.143
Meng, Y., Ban, A. (2024). Automated UML Class Diagram Generation from Textual Requirements Using NLP Techniques. JOIV: International Journal on Informatics Visualization. 8. 1905. DOI: 10.62527/joiv.8.3-2.3482
Merzouk, S., Jabir, B., Marzak, A., Sael, N. (2024). Best Agile method selection approach at workplace. Bulletin of Electrical Engineering and Informatics. 13. 1868-1876. DOI: 10.11591/eei.v13i3.5782
Mesjasz, C., Bartusik, K., Małkus, T., & Sołtysik, M. (2022). Agile Project Management and Complexity: A Reappraisal (1st ed.). Routledge. DOI: 10.4324/9781003175032
Microsoft, "Azure DevOps Services," Microsoft Azure. [Online]. Available: https://azure.microsoft.com/en-us/products/devops. [Accessed: Apr. 9, 2025]
Mohammad, A., Kollamana, J. M. (2024). Causes and Mitigation Practices of Requirement Volatility in Agile Software Development. Informatics, 11(1), 12. DOI: 10.3390/informatics11010012
Nagoya, F. (2021). A Case Study on Combining Agile Requirements Development and SOFL. In: Xue, J., Nagoya, F., Liu, S., Duan, Z. (eds) Structured Object-Oriented Formal Language and Method. SOFL+MSVL 2020. Lecture Notes in Computer Science, vol 12723. Springer, Cham. DOI: 10.1007/978-3-030-77474-5_2
Neto, A. S., Ramos, F., Albuquerque, D., Dantas, E., Perkusich, M., Almeida, H., Perkusich, A. (2023). Towards a Recommender System-based Process for Managing Risks in Scrum Projects. In Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23). Association for Computing Machinery, New York, NY, USA, 1051–1059. DOI: 10.1145/3555776.3577748
Nikiforova O., El Marzouki N., Gusarovs K., Vangheluwe H., Bures T., Al-Ali R., Iacono M., Esquivel P.O., Leon

F. (2017) The two-hemisphere modelling approach to the composition of cyber-physical systems. Proceedings of the 12th International Conference on Software Technologies (ICSOFT), pp. 286 - 293, DOI: 10.5220/0006424902860293

Nikiforova, O., Babris, K., Karlovs-Karlovskis, U., Narigina, M., Romanovs, A., Jansone, A., Grabis, J., & Pastor, O. (2025a). Model Transformations Used in IT Project Initial Phases: Systematic Literature Review. Computers, 14(2), 40. DOI: 10.3390/computers14020040

Nikiforova, O., Babris, K., Miļūne, M. K., Tanguturi, N. and Pastor, Ó. (2025). Key Artefacts in the Initial Phases of IT Project Management: Systematic Mapping Study. In Proc. of the 20th International Conference on Evaluation of Novel Approaches to Software Engineering ENASE; SciTePress, pages 773-781. DOI: 10.5220/0013471000003928

Nikiforova, O., Nikulsins, V., Sukovskis U. (2009) Integration of MDA framework into the model of traditional software development, Frontiers in Artificial Intelligence and Applications, 187 (1), 229 - 239, DOI: 10.3233/978-1-58603-939-4-229

Nikiforova, O., Pavlova, N. (2009) Application of BPMN instead of GRAPES for Two-Hemisphere Model Driven Approach // Advanced Database and Information Systems, Grundspenkis J. et al. (Eds.), Springer, LNCS, pp. 185-192

Noël, R.; Panach, J.I.; Ruiz, M.; Pastor, O. Stra2Bis: A Model-Driven Method for Aligning Business Strategy and Business Processes. In Conceptual Modeling. ER 2022. Lecture Notes in Computer Science; Ralyté, J., Chakravarthy, S., Mohania, M., Jeusfeld, M.A., Karlapalem, K., Eds.; Springer: Cham, Switzerland, 2022; Volume 13607. DOI: 10.1007/978-3-031-17995-2_18.

Noreika, K., Gudas, S. (2021). Using Management Transaction Concept to Ensure Business and EAS Alignment in an Agile Environment. In A. Lopata, et al. (Eds.), Information and Software Technologies (Vol. 1486, 109–120). Springer. DOI: 10.1007/978-3-030-88304-1_9

Pastor, O.; Molina, J. (2007). Model-driven architecture in practice: A software production environment based on conceptual modelling. Springer, DOI: 10.1007/978-3-540-71868-0.

Pastor, O.; Nöel, R.; Panach, I. (2021) From Strategy to Code: Achieving Strategical Alignment in Software Development Projects Through Conceptual Modelling. Transactions on Large Scale Data Knowledge Centred Systems. 48: 145-164, DOI: 10.1007/978-3-662-63519-3_7

Pastor, O.; Noël, R.; Panach, J.I.; Ruiz, M. The LiteStrat Modelling Method: Towards the Alignment of Strategy and Code. In Domain-Specific Conceptual Modeling; Karagiannis, D., Lee, M., Hinkelmann, K., Utz, W., Eds.; Springer: Cham, Switzerland, 2022; pp. 141–159. DOI: 10.1007/978-3-030-93547-4_7.

Pasuksmit, J., Thongtanunam, P., & Karunasekera, S. (2021). Towards Just-Enough Documentation for Agile Effort Estimation: What Information Should Be Documented? IEEE International Conference on Software Maintenance and Evolution, 114–125. DOI: 10.1109/ICSME52107.2021.00017

PMI. (2021). The Standard for Project Management and a Guide to the Project Management Body of Knowledge (7th ed.). PMBOK Guide, Project Management Institute (PMI). ISBN: 978-1-62825-664-2

Prasetyo, M. L., Peranginangin, R. A., Martinovic, N., Ichsan, M., Wicaksono, H. (2025). Artificial intelligence in open innovation project management: A systematic literature review on technologies, applications, and integration requirements. Journal of Open Innovation: Technology, Market, and Complexity, Volume 11, Issue 1, 100445, ISSN 2199-8531, DOI: 10.1016/j.joitmc.2024.100445

Rodríguez Sánchez, E., Vázquez Santacruz, E. F., & Cervantes Maceda, H. (2023). Effort and Cost Estimation Using Decision Tree Techniques and Story Points in Agile Software Development. Mathematics, 11(6), 1477. DOI: 10.3390/math11061477

Sayeb, K., Bhiri, O., Ghannouchi, S. A. (2024). Toward integrating reuse approach within scrum process. 2024 IEEE 7th International Conference on Advanced Technologies, Signal and Image Processing (ATSIP), Sousse, Tunisia, pp. 518-523, DOI: 10.1109/ATSIP62566.2024.10639040

Schlutter, A., Vogelsang, A. (2020). Knowledge Extraction from Natural Language Requirements into a Semantic Relation Graph. In Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops (ICSEW'20). Association for Computing Machinery, New York, NY, USA, 373–379. DOI: 10.1145/3387940.3392162

Sholiq, S., Sarno, R., Astuti, E. S. (2022). Generating BPMN diagram from textual requirements. Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 10, Part B, Pages 10079-10093, ISSN 1319-1578, DOI: 10.1016/j.jksuci.2022.10.007

Talib. R., Kashif, M., Ayesha, S., Fatima, F. (2016). Text Mining: Techniques, Applications and Issues. International Journal of Advanced Computer Science and Applications(ijacsa), 7(11), DOI: 10.14569/IJACSA.2016.071153

Thota, S. R., Arora, S., Gupta, S. (2024). Al-Driven Automated Software Documentation Generation for Enhanced Development Productivity. 2024 International Conference on Data Science and Network Security (ICDSNS), Tiptur, India, pp. 1-7, DOI: 10.1109/ICDSNS62112.2024.10691221

Wagner T. J., Ford T. C. (2020). Metrics to Meet Security & Privacy Requirements with Agile Software Development Methods in a Regulated Environment. 2020 International Conference on Computing, Networking and Communications (ICNC), Big Island, HI, USA, pp. 17-23, DOI: 10.1109/ICNC47757.2020.9049681