Experiences from Combining Merode and Scrum

Yaimara Granados [1], Monique Snoeck [2] , Jenny Ruiz[3] and Gheisa Ferreira [4]

[1] *Central University of Las Villas, Camajuaní Km 5 1/2, Villa Clara, 1, Cuba*
[2] *KU Leuven, Naamsestraat 69, 3000 Leuven, Belgium*
[3] *University of Holguín, Ave XX Aniversario, Vía Guardalavaca Piedra Blanca, Holguín, 7, Cuba*
[4] *Central University of Las Villas, Camajuaní Km 5 1/2, Villa Clara, 3, Cuba*

Abstract

This paper describes the implementation of Agile Merode in the development of a curriculum management system at the Central University of Las Villas (UCLV). The project is carried out in a leading software development company in Cuba, in collaboration with UCLV. Considering the constraint of imposed implementation technologies, and starting from Scrum as ASD method, Agile Merode was used for the generation of functional prototypes at the start of each sprint to validate the requirements of stakeholders.While not used for the generation of final application code, this allowed the increments to be accepted by the stakeholders without significant changes to the progress of the project. The implementation of the proposal allowed to identify, through the observation and follow-up of the project, some challenges that the use of model-driven approaches in agile software development environments presents for the Cuban software industry. We define future work aimed at solving these and other challenges, thus setting a first step in developing a new approach to developing software in Cuban industry.

Keywords

Rapid prototyping, agile software development, model-driven engineering, conceptual modeling, MERODE, Agile Merode

1. Introduction

The research presented in this paper stems from one of the actions carried out as part of the VLIR-UOS South Initiative project "Better digital services for Cuba through Model Driven Engineering". During the workshop "Experiences in Software Development. University and Company Network", held on June 30th in Villa Clara, Cuba, the MERODE method was presented as an alternative for software development and experiences were shared by the invited experts. Representatives from leading software development companies in the country participated and as a result of the debate and the opinions of the experts, some ideas were collected.

- Software development in Cuba has left behind traditional methods and uses agile approaches such as Scrum, XP, UCI-AUP (Adaptation of Agile Unified process in University of Informatics Sciences) and where a methodology is not established, the experts related the practices to the use of Kanban boards.
- The greatest challenge to meeting agreed delivery times is associated with the constant changes in software requirements.
- As an way to mitigate this challenge, the workshop experts suggested using MERODE to create a prototype that users can test before starting to code.

This last idea is the trigger for the present research, which presents challenges in software development related to the introduction of MDE elements in an agile environment. This has been done through a case study in which the MDE-based MERODE method is applied to the development of a Curriculum

Agil-ISE24: 3rd Intl. Workshop on Agile Methods for Information Systems Engineering, June 3, 2024, Limassol, Cyprus
EMAIL: ygranados@uclv.edu.cu (A. 1); monique.snoeck@kuleuven.be (A. 2); jruizp@uho.edu.cu (A. 3);gheisa@uclv.edu.cu (A. 4)
ORCID: 0000-0002-2844-3677 (A. 1); 0000-0002-3824-3214 (A. 2); 0000-0002-1371-6353 (A. 3) ; 0000-0003-1097-0847 (A. 4)

Management System to test customer requirements. The system was developed in collaboration between XETID, a leading development software in Cuba, and the Central University "Marta Abreu" of Las Villas (UCLV). The software project addresses the problem of multiple changes in the curriculum of different educational programs that arose during the pandemic. The result and benefit expected from this work is around the implementation of an MDE method in Cuban software project. The description of the challenges identified in the process are translated as an opportunity for future work. This is the main contribution of the work presented.

The contents of this paper are as follows: Section 2 provides an overview of agile software development, including related work on MDE combined with agile approaches, and presents the MERODE method used in an agile environment. Section 3 describes the development of a curriculum management system using an adapted version of Agile-Merode. Section 4 describes the experience of applying MERODE in an agile environment, both the positive elements and the challenges identified in the process. Section 5 outlines future research and expected results, and section 6 concludes the paper.

2. Background

The Agile Software Development (ASD) approach is presented as a solution to the challenges of constantly changing requirements that cause delays and even project failures at advanced stages of development. ASD also addresses the challenge of rapid software delivery by delivering software in increments to meet the agreed time to market. It has benefits for teamwork, particularly by providing opportunities to increase team productivity. ASD is typically iterative and incremental: a set of requirements is defined and implemented to produce product releases in short periods of time, called product increments [1]. One of the key strengths of ASD is the autonomy of the development teams, who are able to achieve their goals by working as a unit and considering stakeholders as part of the team [2]. Globally speaking, the most common and recognized trend in development teams is the use of agile methodologies such as XP [3], XP-Scrum hybrid [4], Kanban [5], Scrumban [6], Scrum [7], the latter being one of the most popular approach according to State of Agile report 2022 [8]. The acceptance of Scrum in software development environments is due to the fact that it is more than a software methodology: it is a philosophy of work. It is user-centered, gives fast feedback to stakeholders, generates fidelity and commitment to the process. Its major advantage is the sprint planning, defined by the sprint backlog, sprint goals, product release and feedback in short periods of time, between one and four weeks maximum. This approach is thinking for projects where requirement are highly volatile, and rapid change respond is strongly needed. It also focuses on creating a good working environment. The role and ceremonies definition allows project success promoting team autonomy [9]. Although Scrum is required by company policy for this work, the most important agile values that Scrum fulfills for us are customer satisfaction and rapid delivery of valuable software.

2.1. Existing work on modeling + Agile

Model-driven engineering (MDE) encourages the use of models to guide development through different phases of model transformations and code generation. The models provide a high-level perspective of the software being developed and pursue abstraction of platform complexity [10]. Nowadays, this paradigm has gained importance [11] as it enables the creation of a more efficient software engineering process, resulting in higher quality software. Some authors propose frameworks with support tools (Computer Aided Software Engineering, CASE)[10] that allow the transition from computation-independent models, in which essential business concepts promote the understanding and verification of requirements in early stages of development to high-level models that are transformed into platform-specific models or code to support automatic code generation. Examples are: RESTVSoaML [12], FRAMEWEB [13], IADev [14], ORSA [15], MERODE [16], [17] and JMDA [18]. Model-driven development allows investing significantly more effort in model design as the required effort for coding is significantly reduced.

Although MDE-based approaches have several advantages and can be used to reduce complexity there is still room for improving their use in the software development industry. It has been challenging to incorporate a systematic use of such approaches in practice [19]. While ASD focuses on the user and

the time to respond to requirement changes, the architecture aspect is not strongly considered [20]. In contrast, MDE is based on creating models to achieve a strong, extensible and scalable software architecture [21], but it suffers from a lack of user orientation. Although combining conceptual models with agile approaches is not a new idea, this topic has recently received special attention [22]. One of the most common arguments for using conceptual models in ASD is to improve communication and understanding of user requirements, which helps in their validation and verification [23]. Also, the benefits identified in [24] such as increasing team productivity, software quality, and customer satisfaction, support the combination of MDE with agile. However, the effort of creating models can be in conflict with agile values, which is a challenge in agile software development [25], [26]. In [27] a new role and task in agile approaches is proposed: the role of domain expert tasked with the representation of requirements through business process models [28]. Other studies such as [29], [30] share the idea of using models to represent requirements, but rather extend the set of models to be used with story cards and mind maps. Recently, [31] proposed to automatically generate use case models, the domain model and state charts from users stories and Behavior Driven Development (BDD) scenarios to improve communication and understanding of the requirements. None of the above approaches truly is model-driven in the sense that the models are used for improving some aspect of the software development process (e.g. requirements understanding), but not to generate software from them.

2.2. Merode & Agile-MERODE

MERODE is an MDE methodology that enables the specification of an enterprise system using a conceptual domain model. The developed model is platform-independent and at the same time complete enough to automatically generate fully functional system code from it [17]. The model used by MERODE is based on a UML class diagram that captures the domain classes (business object type), an object-event table (OET) that captures interaction aspects, and finite state machines (FSM) that describe the behavior of domain objects. The MERODE method has its own proven conceptual modeling environment, Merlin, which allows rapid prototyping of a conceptual domain model. This tool has an MDE-based code generator that generates a fully functional Java prototype application from a MERODE conceptual model, and this supporting tool has been successfully tested and validated for teaching conceptual modeling for more than 10 years [32].

The MERODE method was originally designed for non-agile environments, but it is possible to adapt it to agile environments through the Agile-MERODE specification proposed in [33]. MERODE supports iterative incremental development, as it is possible to generate an application from a model containing a single business object type first, and then expand it with additional business object types. It facilitates the management of development through sprint planning constraints derived from the existence of dependencies. In general, business object types that do not require the existence of other objects for their own existence allow the baseline of the architecture to be defined. This is one of the strengths that keeps the focus on robust architecture development.

The Agile-MERODE framework therefore proposes to perform domain modeling in parts. It is based on composite user stories through Behavior Driven Development (BDD) scenarios, where the who (role), what (goal), and even what (benefit) are defined by specifying the context, the user behavior, and the expected results [33]. Agile-MERODE suggests planning a Sprint 0 during which the business object types are identified and a domain model is created. In Sprint 0, the initial domain model is defined and a functional prototype is generated using the MERODE CASE tools. This first prototype allows a first feedback on the desired features to be realized in the next sprint. The next sprints receive as input the domain model produced in the previous sprint. When all sprints have been completed, the resulting model responds to all requirements of the system [33].

3. Case study

In this section, we present the case study during which domain modelling and MDE were combined with Scrum as ASD. We first present the project, and then the adaptations that were done to the Agile-MERODE method.

3.1. The project: Curriculum Management

The System for Curriculum Management at the Central University "Marta Abreu" of Las Villas (UCLV) has its origins in the pandemic, when it was necessary to revert to e-learning and to adjust the planning of the courses to this. The epidemiological conditions constrained the planning of lectures, labs and practice sessions, field work, and presential evaluations. All these constraints, in combination with other concerns from students and teachers made the teaching process quite complex.

The UCLV is a complex campus with many multidisciplinary courses. In Cuba, all changes to a curriculum must be legalized beforehand, and the Ministry of Higher Education (MES) adjusted the rules in response to the pandemic. The University, in order to comply with the new Higher Education Regulations issued by the MES, worked hard to allow the continuity of the academic year. The process to legalize all the proposed changes was exhausting and the consumption of resources was excessive, considering that each cycle of changes was repeated at least twice. Thus, the university management was immersed in another crisis, a crisis of resources.

This experience led to the development of a system that would support the management of study plans, as well as the review and approval processes for proposals, and in which the legal authority would take a key role. This first version of UCLV's Curriculum Management System was developed collaboratively by XETID and UCLV in an effort to overcome the difficulties encountered in the past and as a first step towards digital transformation by promoting the digitization of university processes and legal documents. The input to the system is a request for modification of the study plan, submitted by head of the department and which specifies the courses and each of the changes to be made according to the defined legal bases. In addition, for each change request, it is necessary to attach a scan of the currently approved study plan and of the proposed study plan with an indication of the changes to the teaching process. This was a significant change in the way study plan changes were processed and after a while, the use of the system revealed gaps that required further improvement of the solution.

The system was developed using the Process Maker platform, which is good at document management, but lacks a good support for the business logic that governs data and information processes. On the positive side, it stands out for its reliability in handling documents and digital signatures: it allows modelling business processes including the complete document cycle, e.g. to control author manipulation, generate complementary documents in a process, and design review-approval- rejection flows. Another advantage is the traceability of documents, of the people who manipulated them, the dates of changes, as well as knowing at all times the status and place in the process where a given request is located. Despite these benefits, the system does not directly include curriculum management features, which means that mistakes are often made when submitting change requests and a document is passed through the same hands multiple times. This delays the approval of a curriculum change and significantly affects the user experience. It was therefore proposed to develop a new version of the system that includes functionalities for 1) creating new version of study plans, 2) Modifying courses and exams typology, 3) storing the current proved version of study plan, 4) Managing the legality of each permitted changes to a curriculum, and 5) generating formal models to represent modified study plan

3.2. Development environment

The project followed the Scrum guide. The development team consisted of seven members: the Scrum Master, the Product Owner [1] and five team members. The Product Owner was part of the development team and was involved in the whole process from the beginning. We used the technology defined as policy in the company: Laravel (backend), Vue (frontend), and Process Maker (document cycle management). The potential users of the system are department heads, deans, methodological advisors, the vice rector of professional training and the head of the legal department of the UCLV. About seventy people use and benefit from the system. In this case, the end user is the Director of the Methodology Department of the Vice-Rector's Office for Professional Training at UCLV, who became an active part of the team project due to his strong interest and commitment to the work result.

Most Scrum development teams define requirements through user stories [34]. A user story is an informal text that describes wishes (requirements) based on the product owner's vision. Behavioral scenarios have been included in user stories to better understand and obtain more formal definitions of system requirements [35]. The product backlog [36], [37] was created with important features, collected to give a minimum viable product (MVP). The user stories were written using the following metadata: identifier, name, description, priority, and acceptance criterion, as well as an estimate based on previous experience.

Table 1 shows an extract of the product backlog. In total, fifty-one user stories were defined and implemented. The planning of each sprint was guided by the architectural priority and the similarity in the implementation of the functionality. Table 2 shows the four planned sprints, the defined goals, and the description of each goal. Since the first sprint was planned, we worked with sprint backlogs. The prioritization of the sprint backlog was done by the product owner, who has extensive knowledge of the business domain and has participated in previous developments with the same team.

Table 1
Product backlog extract

ID	User story
US0001	Register Study Center
US0002	Register Faculty
US0003	Register your department
US0004	Register Training Program
US0005	Register academic years
US0006	Register Professor
US0007	Register Role
US0008	Assign responsibility
US0010	Register subject
US0011	Register curriculum type
US0012	Register Discipline
US0013	Configure Exam Type
US0014	Register legal basis
US0015	Configure curriculum
US0016	Register New Curriculum Amendment
US0017	Assign legal basis
US0018	Generate Appendix 1 according to curriculum changes
US0019	Generate MC11-C

Table 2
Definition of sprints

Sprint	Sprint goal	Description
1	Management of basic entities	Implementation of functions on entities that define the system structure. For example, Faculty, Department, Career, Academic Year.
2	Management of curriculum entities	Implementation of the entities that make up the curriculum of a course. E.g. type of curriculum, discipline, subjects, typology of final evaluations
3	Curriculum configuration	Creation, modification, cancellation of a curriculum and its versions.
4	Integration with Process Maker platform	Configuration and generation of legal documents for the institution. Example: Appendix 1, Appendix 3, Model MS11-C.

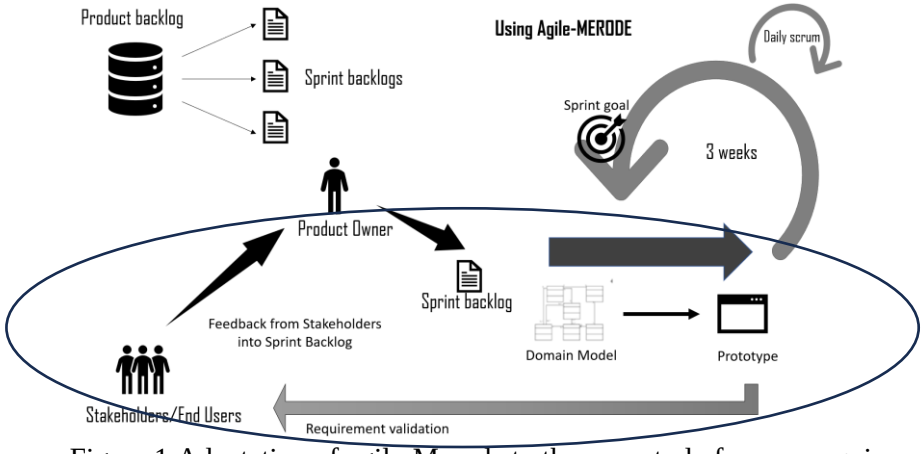

Figure 1:Adaptation of agile-Merode to the case study for user requirement validation.

3.3. Adaptation of the method

Agile-MERODE was applied to the planning of each sprint to validate user requirements before coding begins, so the functioning of Scrum was modified and before starting the cycle of each sprint it was necessary to perform a preliminary cycle as shown in Figure 1. The preliminary cycle would consist of creating a MERODE domain model at the beginning of each sprint based on the user's partial requirements, and automatically generating the functional prototype to show stakeholders a version of how their requirements will be implemented. Based on stakeholder feedback, the modeling and prototyping cycle is repeated until the model is stable.

From a sprint backlog formed by a set of previously selected user stories, the business object types were identified, and Agile-MERODE [33] was executed up to the point of automatically generating the functional prototype corresponding to the features of the sprint. At this point, the prototype is shown to and analyzed by the stakeholders, who, taking into account the concepts created and the precedence between them, validate that the development team has understood their needs and that the requirements have been correctly handled in the system. If the feedback is positive, the (manual) implementation begins according to the developed domain model that defines the architectural baseline. If the feedback reveals problems, the sprint backlog and the product backlog are updated with the changes introduced by the stakeholders. For example, in Sprint 1, the prototype was built from the UML class diagram shown in Figure 2. This model needed to change, and the prototype was built twice to incorporate new features and modify the initial design before implementation began as a result of stakeholder feedback.

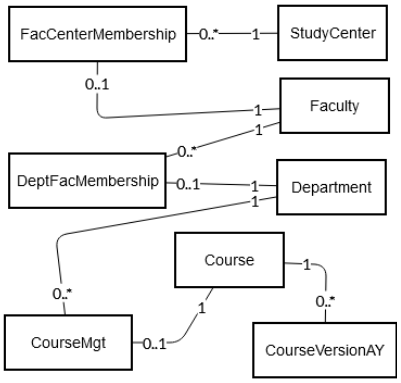

Figure 2: UML CD for Sprint 1

4. Evaluation/Lessons Learned

4.1. Positive experiences

The application of Agile-MERODE in the project shows that it is possible use MDE without compromising agile values. The use of MERODE allowed the creation of a functional prototype at the beginning of each sprint, which encouraged active communication between stakeholders and the development team. This resulted in a reduced number of requirement changes in later sprints. One of the main benefits experienced was that, thanks to the cycle of fast prototyping at the beginning of each sprint, stakeholder feedback at the end of a sprint did not lead to relevant code changes. In addition, the process was able to respect agile values, and even though each sprint required some modelling time to start with, deadlines were met. The stakeholders' satisfaction with the generated prototype was a positive experience and contributed to their feeling of being part of the team. We believe that the idea of introducing a preliminary step focused on modeling at the beginning of the sprint was successful in terms of understanding user requirements. It was possible to develop the domain model incrementally, without delays. The difference with other proposals to combine MDE with agile approaches was the positive impact on the requirement engineering steps, which enabled stronger communication between stakeholders and the project team through the generation of rapid prototypes.

4.2. Challenges

Agile-MERODE has been adopted only in part, to respect the use of mandatory implementation technologies. Although the application of Agile-MERODE was considered a positive experience by the team members, some challenges were observed. The use of Agile-Merode led to increased effort due to the learning curve for Merode method and its tools. It took time to change the way requirements were handled. After specifying user stories for each sprint backlog, we developed a UML CD model for

prototyping, requiring additional effort and development of analyst skills in modeling and prototyping. This confirms the challenges identified in [24] of a high learning curve for MDE and a lack of experience in the Cuban industry on this field. The challenge demonstrates the importance of incorporating the systematic use of conceptual models as a best practice for requirements validation and architecture baselining into the training of analysts. Another challenge was validating the integrity of the conceptual model obtained as a result of the partial models of each sprint backlog. The most common mistake was the duplication or omission of business concepts when showing the prototype to stakeholders. We believe this comes from working in parts as a result of developing partial models for each sprint. The model completeness assessment is necessary when new functionality is added in a new sprint.

5. Future works

This field of combining Agile and modelling is very wide and many different directions of research are possible. To scope the further research, our proposal intents to combine the best practices of recent studies to solve the identified challenges in previous section and investigate generalization to other projects in other Cuban companies. The design and implementation of a proposal for the Cuban software industry, towards improvement of software development in agile environment will be addressed by the following research:

Survey on willingness to use models and perceived utility of models by agile teams: The survey is targeted to both practitioners and students. For practitioners is important to obtain their positions as a starting point to measure the existing perception of the use of models in agile software development. The students are a future development team members. It would be valuable to understand their perception of modeling in software development from their academic experience. These results can be interesting to create and systematize modeling habits.

Review of model generation approaches from natural language and structured text: This review will represent the state of the art of generation of concepts from natural language. Resents studies go for generating conceptual models like use case diagrams, class diagrams, state chart diagrams from textual requirement specifications ([31], [38], [39], [40]). In contrast with our approach, these proposals generate models from the user stories, whereas the approach in our project relied on manual modelling. Future work can research whether model generation approaches can further minimize the time spent to modelling, and which set of approaches are applicable to our scope. Several indicators would be identified to select approaches and evaluate the quality of the generated models. This research is a support activity to respond to the first challenge and may result in the design of an alternative to manual modeling, thus contributing to the reduction if the effort of building conceptual models from requirements.

Enhanced Structured Text Format (BDD+): To facilitate the transformation of requirements into models, we will research an enhanced BDD+ format for specifying the requirements. The proposed design would aim for a better translation of natural language to models through NLP techniques, allowing the NLP algorithms to better identify important business concepts. This goal answers to the second challenge. The improvement of the format to specify user requirements should also positively impact in the integrity of the develop models.

NLP support to move from natural language requirements to BDD+: To further facilitate the formulation of requirements, we intend to research the use of NLP to facilitate the formulation of requirements in the BDD+ format.

The different research tracks will be applied in projects to validate the proposed solutions and measure their utility and/or readiness to use. We intend to validate the value of the approach for developers in the Cuban software industry and further extend the validation towards the increase of team productivity as a fundamental agile value.

6. Conclusions

This research in this paper has been conducted through a case study that presents an example of the use of MDE in an agile environment in Cuba software industry. The described case study was Scrum-based development, and the sprint planning was modified. We introduced the Merode artifacts (EDG, OET, FSM) and prototype generation at the beginning of each sprint, following the Agile-Merode

approach. The experiment was considered satisfactory and good experiences were found. The most important experience was the customer's satisfaction with a rapid prototyping of their requirements. We identified challenges from the execution of a case study and we propose research projects to address these challenges and further improve the software development practice.

7. Acknowledgements

The research was funded by the VLIR-UOS South Initiative project “Better digital services for Cuba through Model Driven Engineering”, code number CU2022SIN332A101.

8. References

[1] M. D. Kadenic, D. A. de Jesus Pacheco, K. Koumaditis, G. Tjørnehøj, and T. Tambo, “Investigating the role of Product Owner in Scrum teams: Differentiation between organisational and individual impacts and opportunities,” *Journal of Systems and Software*, vol. 206, Dec. 2023, doi: 10.1016/j.jss.2023.111841.

[2] M. Guerrero-Calvache and G. Hernández, “Team Productivity Factors in Agile Software Development: An Exploratory Survey with Practitioners,” in *Communications in Computer and Information Science*, Springer Science and Business Media Deutschland GmbH, 2024, pp. 261–276. doi: 10.1007/978-3-031-46813-1_18.

[3] C. J. Stettina, J. Garbajosa, and P. Kruchten, *Agile Processes in Software Engineering and Extreme Programming: 24th International Conference on Agile Software Development, XP 2023, Amsterdam, The Netherlands, June 13–16, 2023, Proceedings*. Springer Nature, 2023.

[4] Z. Mushtaq and M. R. J. Qureshi, “Novel hybrid model: Integrating scrum and XP,” *International Journal of Information Technology and Computer Science (IJITCS)*, vol. 4, no. 6, p. 39, 2012.

[5] H. Alaidaros, M. Omar, and R. Romli, “The state of the art of agile kanban method: challenges and opportunities,” *Independent Journal of Management & Production*, vol. 12, no. 8, pp. 2535–2550, 2021.

[6] J. Fuentes Del Burgo, S. Pérez, and M. Ángel, “Comparative analysis of the board tool in the agile methodologies scrum, kanban and scrumban in software projects,” in *26 th International Congress on Project Management and Engineering Terrassa*, 2022.

[7] K. Beek, G. Wagenaar, L. Kester, S. Overbeek, and E. de Rooij, “Measuring Team Effectiveness in Scrum,” 2023, pp. 233–247. doi: 10.1007/978-3-031-43703-8_17.

[8] “State of Agile Report,” 2022.

[9] S. Sachdeva, “Scrum Methodology,” *International Journal Of Engineering And Computer Science*, Jun. 2016, doi: 10.18535/ijecs/v5i6.11.

[10] D. Hallmann, U. Schmid, and R. Von der Weth, “Shared mental models in the agile software development: An approach to the drafting of design recommendations for ‘good’ experience-specific user stories | Gemeinsame mentale Modelle in der agilen Softwareentwicklung: Ein Ansatz zur Erstellung von Gestalt,” in *Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI)*, 2016, pp. 1969–1974.

[11] A. Bucchiarone, J. Cabot, R. F. Paige, and A. Pierantonio, “Grand challenges in model-driven engineering: an analysis of the state of the research,” *Softw Syst Model*, vol. 19, pp. 5–13, 2020.

[12] A. Kenzi and F. Yakine, “A Model Driven Framework for the Development of Adaptable REST SERVICES,” in *International Conference on Web Information Systems and Technologies, WEBIST - Proceedings*, 2021, pp. 544–552.

[13] P. H. B. Hoppe and V. E. S. Souza, “Support for Single Page Application Frameworks on FrameWeb,” in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Oct. 2023, pp. 260–268. doi: 10.1145/3617023.3617059.

[14] W. Rafique, X. Zhao, S. Yu, I. Yaqoob, M. Imran, and W. Dou, “An Application Development Framework for Internet-of-Things Service Orchestration,” *IEEE Internet Things J*, vol. 7, no. 5, pp. 4543–4556, 2020, doi: 10.1109/JIOT.2020.2971013.

[15] J. Sommer, A. Gerndt, and D. Lüdtke, “Creating a reliable data type framework for the OSRA using modern C++,” in *Proceedings of the International Astronautical Congress, IAC*, 2019.

[16] V. Amaral de Sousa, C. Burnay, and M. Snoeck, “B-MERODE: A Model-Driven Engineering and Artifact-Centric Approach to Generate Blockchain-Based Information Systems,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer, 2020, pp. 117–133. doi: 10.1007/978-3-030-49435-3_8.

[17] M. Snoeck, *Enterprise information systems engineering: The MERODE approach*. 2014.

[18] R. Moreno-Rodríguez, “CASE jMDA de Arquitectura Dirigida por Modelos para Sistemas de Información CASE jMDA for Model Driven Architecture for Information Systems,” *Revista Cubana de Ciencias*

Informáticas, vol. 14, no. 4, 2020, [Online]. Available: http://rcci.uci.cuPág.210-223Editorial"EdicionesFuturo"
[19] C. Verbruggen and M. Snoeck, "Practitioners' experiences with model-driven engineering: a meta-review," *Softw Syst Model*, vol. 22, no. 1, pp. 111–129, 2023, doi: 10.1007/s10270-022-01020-1.
[20] N. Rios, M. G. Mendonça, C. B. Seaman, and R. O. Spínola, "Causes and Effects of the Presence of Technical Debt in Agile Software Projects," in *Americas Conference on Information Systems*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:201114768
[21] N. Kharmoum *et al.*, *Agile User Stories' Driven Method: A Novel Users Stories Meta-model in the MDA Approach*, vol. 637 LNNS. 2023. doi: 10.1007/978-3-031-26384-2_13.
[22] K. Lano, S. Kolahdouz-Rahimi, J. Troya, and H. Alfraihi, "Introduction to the theme section on Agile model-driven engineering," *Software and Systems Modeling*, vol. 21, no. 4. Springer Science and Business Media Deutschland GmbH, pp. 1465–1467, Aug. 01, 2022. doi: 10.1007/s10270-022-01016-x.
[23] A. Gupta, G. Poels, and P. Bera, "Using Conceptual Models in Agile Software Development: A Possible Solution to Requirements Engineering Challenges in Agile Projects," *IEEE Access*, vol. 10, pp. 119745–119766, 2022, doi: 10.1109/ACCESS.2022.3221428.
[24] H. Alfraihi and K. Lano, "The integration of agile development and model driven development: A systematic literature review," in *MODELSWARD 2017 - Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development*, SciTePress, 2017, pp. 451–458. doi: 10.5220/0006207004510458.
[25] V. N. Vithana, "Scrum requirements engineering practices and challenges in offshore software development," *Int J Comput Appl*, vol. 116, no. 22, 2015.
[26] S. Alam, S. N. Bhatti, S. A. A. Shah, and A. M. Jadi, "Impact and challenges of requirement engineering in agile methodologies: A systematic review," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 4, 2017.
[27] M. Daneva *et al.*, "Agile requirements prioritization in large-scale outsourced system projects: An empirical study," *Journal of systems and software*, vol. 86, no. 5, pp. 1333–1353, 2013.
[28] M. Trkman, J. Mendling, and M. Krisper, "Using business process models to better understand the dependencies among user stories," *Inf Softw Technol*, vol. 71, pp. 58–76, 2016.
[29] V. Kannan *et al.*, "User stories as lightweight requirements for agile clinical decision support development," *Journal of the American Medical Informatics Association*, vol. 26, no. 11, pp. 1344–1354, 2019.
[30] E.-M. Schön, J. Thomaschewski, and M. J. Escalona, "Agile Requirements Engineering: A systematic literature review," *Comput Stand Interfaces*, vol. 49, pp. 79–91, 2017.
[31] A. Gupta, "Generation of multiple conceptual models from user stories in agile," in *CEUR Workshop Proceedings*, 2019.
[32] M. Snoeck, "Overview of MERODE," in *Enterprise Engineering Series*, Springer, 2014, pp. 51–75. doi: 10.1007/978-3-319-10145-3_3.
[33] M. Snoeck and Y. Wautelet, "Agile MERODE: a model-driven software engineering method for user-centric and value-based development," *Softw Syst Model*, vol. 21, no. 4, pp. 1469–1494, 2022, doi: 10.1007/s10270-022-01015-y.
[34] B. Yang, X. Ma, C. Wang, H. Guo, H. Liu, and Z. Jin, "User story clustering in agile development: a framework and an empirical study," *Front Comput Sci*, vol. 17, no. 6, 2023, doi: 10.1007/s11704-022-8262-9.
[35] A. Gupta, "Generating Conceptual Models from User Stories: Framework, Algorithm, and Tool," 2023.
[36] T. Sedano, P. Ralph, and C. Peraire, "The Product Backlog," in *Proceedings - International Conference on Software Engineering*, 2019, pp. 200–211. doi: 10.1109/ICSE.2019.00036.
[37] S. Mosser, C. Pulgar, and V. Reinharz, "Modelling Agile Backlogs as Composable Artifacts to support Developers and Product Owners," *Journal of Object Technology*, vol. 21, no. 3, 2022, doi: 10.5381/jot.2022.21.3.a3.
[38] S. Nasiri, Y. Rhazali, M. Lahmer, and N. Chenfour, "Towards a Generation of Class Diagram from User Stories in Agile Methods," in *Procedia Computer Science*, 2020, pp. 831–837. doi: 10.1016/j.procs.2020.03.148.
[39] S. Nasiri, Y. Rhazali, A. Adadi, and M. Lahmer, *Generation of User Interfaces and Code from User Stories*, vol. 400. 2023. doi: 10.1007/978-981-19-0095-2_38.
[40] S. Nasiri, A. Adadi, and M. Lahmer, "Automatic generation of business process models from user stories," *International Journal of Electrical and Computer Engineering*, vol. 13, no. 1, pp. 809–822, 2023, doi: 10.11591/ijece.v13i1.pp809-822.