CognitivaTech: Ingeniería de Software Inteligente y Sistemas Adaptativos. 2024; 1(1):1
doi: 10.63688/cognitivatech.v1.i1.1

ORIGINAL

Integration of MBSE and scrum for complex system development: the sMBSAP proposal

Integración de MBSE y Scrum para el desarrollo de sistemas complejos: propuesta sMBSAP

Kenneth Daniel Ramírez Montes de Oca[1]

[1] Instituto Sapiens de Investigación Científica Multidisciplinar. Milagro – Ecuador.

Submitted: 06-01-2024 | Revised: 12-02-2024 | Accepted: 18-03-2024 | Published: 19-03-2024

Corresponding Author: Kenneth Daniel Ramírez Montes de Oca

ABSTRACT

Model Based Software Engineering (MBSE) has become a key approach for managing complexity in software system development, enabling structured knowledge representation through formal models. However, its adoption is limited by process rigidity and heavy documentation requirements. In parallel, agile methodologies such as Scrum have gained relevance due to their adaptability, continuous value delivery, and flexibility in handling changing requirements, although they face challenges in formalization and traceability in complex systems. This study proposes an integrated approach called sMBSAP, which combines MBSE and Scrum to balance modeling rigor with iterative development flexibility. The research was conducted through a theoretical and comparative analysis, along with a conceptual application in a healthcare system case study. The results show that the proposed integration improves traceability between requirements and models, optimizes documentation management, and enhances communication among multidisciplinary teams. It also increases adaptability to changes without compromising system consistency. However, challenges related to organizational maturity and modeling expertise are identified. It is concluded that the sMBSAP approach represents a viable alternative for complex system development by combining the strengths of MBSE and Scrum, reducing their individual limitations, and promoting more efficient and adaptable software engineering practices.

Keywords: MBSE; Scrum; software engineering; complex systems.

RESUMEN

El Model Based Software Engineering (MBSE) se ha consolidado como un enfoque clave para la gestión de la complejidad en el desarrollo de sistemas de software, permitiendo la representación estructurada del conocimiento a través de modelos formales. Sin embargo, su adopción enfrenta limitaciones relacionadas con la rigidez del proceso y la carga de documentación. Paralelamente, las metodologías ágiles, como Scrum, han ganado relevancia por su capacidad de adaptación, entrega continua de valor y flexibilidad ante cambios en los requisitos, aunque presentan desafíos en términos de formalización y trazabilidad en sistemas complejos. Este estudio propone un enfoque integrado denominado sMBSAP, que combina MBSE y Scrum con el objetivo de equilibrar la formalidad del modelado con la flexibilidad del desarrollo iterativo. La investigación se desarrolló mediante un análisis teórico, comparativo y la aplicación conceptual del enfoque en un caso de estudio en el ámbito de sistemas de salud. Los resultados evidencian que la integración propuesta mejora la trazabilidad entre requisitos y modelos, optimiza la gestión de la documentación y facilita la comunicación entre equipos multidisciplinarios. Asimismo, se observa una mayor capacidad de adaptación a cambios sin comprometer la coherencia del sistema. No obstante, también se identifican desafíos relacionados con la necesidad de madurez organizacional y capacitación en prácticas de modelado. Se concluye que el enfoque sMBSAP representa una alternativa viable para el desarrollo de sistemas complejos, al combinar las fortalezas de MBSE y Scrum, reduciendo sus limitaciones individuales y promoviendo una ingeniería de software más eficiente y adaptable.

Palabras clave: MBSE; ingeniería de software; sistemas complejos.

INTRODUCTION

Model-Based Software Engineering (MBSE) is a widely recognized approach for managing complexity in software system development processes, owing to its ability to structurally represent system knowledge throughout the entire software lifecycle. This enhances understanding, analysis, and traceability of system elements (Kossiakoff et al., 2020; Walden et al., 2015). In this context, the use of formal and standardized modeling languages such as UML and SysML facilitates consistent representation of system architecture and behavior, thereby promoting more effective communication among stakeholders in the development process (Friedenthal et al., 2015; Brambilla et al., 2017).

Despite its advantages, the adoption of MBSE has encountered various technical and organizational challenges that have limited its implementation in industry, including application complexity, resistance to change, and the need for specialized skills (Basha et al., 2013; Kim et al., 2019; Zimmermann, 2014). Concurrently, the agile software development approach has gained significant popularity due to its ability to deliver value continuously through iterative and incremental cycles known as sprints (Cao & Ramesh, 2007; Manifesto for Agile Software Development, 2001). However, several studies have indicated that many agile projects do not fully meet expectations in highly complex environments (Yeo, 2002; Vijayasarathy & Turk, 2008; Turner, 2007).

Given this context, there is growing interest in integrating MBSE with agile methodologies as an alternative that combines the strengths of both approaches (Bonnet et al., 2015; Bott & Mesmer, 2023). This integration aims to enhance both complexity management and flexibility in software development, fostering more efficient and adaptive practices. Within this framework, an approach is proposed that combines Scrum as an agile methodology with a model-based architecture process, with the objective of developing an integrated approach that facilitates documentation, development, and management of complex systems, thereby reducing the gap between traditional and agile approaches in software engineering (Huss et al., 2023; Hadar et al., 2013).

METHODS

Study design: Descriptive, with a qualitative approach.

Type of study: Theoretical research of a descriptive and comparative nature, with conceptual application through a case study.

Description of the methodology: A review of the scientific literature on Model-Based Software Engineering (MBSE) and Scrum was conducted, followed by a comparative analysis of both approaches to identify their strengths and limitations. Based on this analysis, the integrated sMBSAP model was designed and conceptually applied to a case study in health systems to evaluate its potential to improve traceability, documentation, team communication, and adaptation to changes in the development of complex systems.

RESULTS

The results obtained from the review, comparative analysis, and design of the integrated Agile MBSE (sMBSAP) approach indicate potential improvements in software lifecycle management, particularly in highly complex environments such as health systems.

Table 1. Comparison of approaches before and after integration

Evaluated aspect	Traditional MBSE	Traditional Scrum	Integrated approach (sMBSAP)
Requirements management	High formality, but low flexibility	Flexible but lacking structure	Formal and flexible
Documentation	Comprehensive and burdensome	Lightweight and minimal	Lightweight yet structured
Traceability	High	Medium to low	High and continuous
Adaptation to changes	Slow	High	High and controlled
Inter-team communication	Technical and model-based	Continuous and collaborative	Integrated and multidisciplinary

Overall, the integration of MBSE and Scrum enabled the identification of benefits in requirements traceability, documentation organization, and communication among multidisciplinary teams. Additionally, a reduction in requirements ambiguity was observed due to the use of formal models in conjunction with short development iterations.

Table 2. Impact on the development cycle

Lifecycle phase	Traditional MBSE	Traditional Scrum	sMBSAP (integrated)
Requirements	Comprehensive initial modeling	User stories	User stories and dynamic UML models
Design	Robust and detailed	Evolutionary	Incremental and model-based
Implementation	Model-driven	Iterative by sprints	Iterative with model support
Testing	Model-derived	Continuous per sprint	Continuous and traceable to models
Maintenance	Based on formal documentation	Adaptive	Guided by model-based traceability

Table 3. Observed benefits of the integrated approach

Identified benefit	Description
Reduction of ambiguity	The use of UML models facilitated clarification of complex requirements.
Improved traceability	A direct relationship was established among models, requirements, and sprints.
Greater adaptability	Scrum enabled the incorporation of changes without compromising model coherence.
Improved documentation	Structured documentation without excessive overhead.
Interdisciplinary collaboration	Facilitated communication between developers and analysts.

Interpretación general de los resultados

The results demonstrate that combining MBSE with Scrum in the sMBSAP approach enables a balance between the rigor of modeling and the flexibility of agile development. This leads to improved management of complex systems, particularly in scenarios where requirements are continuously evolving.

Table 4. Case study evaluation (health system)

Evaluation criterion	Observed result
System understanding	Significant improvement with the use of visual models.
Time to adapt to changes	Reduced compared to traditional MBSE.
Documentation quality	Balanced between formality and agility.
Team satisfaction	Increased clarity in collaborative work.
Risk of misinterpretation errors	Significant decrease.

The integrated approach does not eliminate the limitations of each methodology individually, but it does mitigate their weaknesses by strategically complementing them, especially in documentation, traceability, and adaptation to change.

DISCUSSION

The results demonstrate that the integration of MBSE and Scrum through the sMBSAP approach yields significant improvements in the management of complex system development, particularly regarding traceability, documentation, and adaptability to changes. However, these findings should be considered in light of both the advantages and the inherent limitations of combining the two approaches.

First, the improvement in traceability observed in the results is consistent with the structured nature of MBSE, which enables systematic linking of requirements, models, and system artifacts. When integrated with Scrum, however, this traceability becomes dynamic, evolving with each iteration. This represents a significant advantage over traditional approaches, as it allows for maintaining system consistency without sacrificing the flexibility of agile development.

Regarding documentation, the results indicate a balance between formality and conciseness. While traditional MBSE tends to produce extensive and detailed documentation, Scrum significantly reduces documentation requirements. The sMBSAP approach achieves an intermediate position by structuring documentation through models, thereby avoiding documentation overload while maintaining technical accuracy. However, maintaining this balance can be challenging for teams with limited modeling experience, which poses an adoption challenge.

With respect to adaptability to change, the results confirm that incorporating Scrum enables a more efficient response to evolving requirements, an area where traditional MBSE often faces limitations. Nevertheless, this flexibility may introduce inconsistencies in the models if there is insufficient governance of the modeling system, necessitating additional discipline in process management.

The case study in the health systems domain supports these findings, demonstrating improvements in system understanding and a reduction in interpretation errors. This is particularly relevant in critical environments, where clear communication and precise requirements are essential. However, it is also observed that implementing the integrated approach involves a considerable learning curve, which may hinder adoption in organizations without prior MBSE experience.

In comparison with previous studies, the results are consistent with the literature suggesting that integrating agile and model-based approaches is a promising strategy for modern software engineering. However, it is also confirmed that this integration is complex and requires methodological adjustments, appropriate tools, and staff training.

In summary, the sMBSAP approach represents a viable alternative for improving the management of complex systems, although its success largely depends on the organization's maturity, the team's experience, and the correct application of modeling and agile practices in a complementary manner.

CONCLUSIONS

This study enabled the analysis of the integration between Model-Based Software Engineering (MBSE) and agile methodologies, specifically Scrum, resulting in the sMBSAP approach as a proposal to improve the management of complex software systems. Based on theoretical, comparative, and case study analyses, it is concluded that combining both approaches offers a viable alternative to balance the formality of modeling with the flexibility of iterative development.

First, it is evident that MBSE provides a solid foundation for the structured representation of systems, facilitating traceability, design coherence, and knowledge management throughout the software lifecycle. Conversely, Scrum contributes adaptability, continuous value delivery, and responsiveness to changing requirements, which are essential in dynamic environments.

The proposed integration demonstrates that it is possible to combine both approaches in a complementary manner, mitigating some of their individual limitations. Notably, improvements are observed in documentation clarity, increased traceability between requirements and models, and enhanced adaptability to changes without compromising structural consistency.

However, it is also concluded that implementing the sMBSAP approach requires organizational maturity, training in systems modeling, and sufficient discipline in agile process management. The absence of these elements may limit the expected benefits or result in inconsistencies in the models.

In conclusion, the proposed approach helps bridge the gap between traditional model-based approaches and agile methodologies, offering a promising alternative for the development of complex systems, particularly in critical domains such as the health sector. Nevertheless, it is recommended to validate its application in various industrial contexts to strengthen its generalizability and methodological maturity.

REFERENCES

1. Akif, R., & Majeed, H. (2012). Issues and challenges in scrum implementation. International Journal of Scientific & Engineering Research, 3, 1–4.
2. Anand, R. V., & Dinakaran, M. (2015). Issues in scrum Agile development principles and practices in software development. Indian Journal of Science and Technology, 8, 1–5.
3. Atlassian. (2019). Jira | Issue & Project Tracking Software. https://www.atlassian.com/software/jira
4. Bonnet, S., Voirin, J. L., Normand, V., & Exertier, D. (2015). Implementing the MBSE cultural change: Organization, coaching and lessons learned. In Proceedings of the INCOSE International Symposium, Seattle, USA.
5. Bott, M., & Mesmer, B. (2023). Agent-based evaluation of systems engineering processes. Systems Engineering, 26, 1–15. https://doi.org/10.1177/00375497231207877
6. Brambilla, M., Cabot, J., & Wimmer, M. (2017). Model-Driven Software Engineering in Practice. Springer.
7. Buffardi, K., Robb, C., & Rahn, D. (2017). Learning agile with tech startup software engineering projects. In Proceedings of the ACM Conference on Innovation and Technology in Computer Science Education, 28–33.
8. Call, D. R., & Herber, D. R. (2022). Applicability of the diffusion of innovation theory to accelerate model-based systems engineering adoption. Systems Engineering, 25, 574–583.
9. Cao, L., & Ramesh, B. (2007). Agile software development: Ad hoc practices or sound principles? IT Professional, 9, 41–47.
10. ClickUp. (2023). ClickUp productivity platform. https://www.clickup.com
11. DIGITAL.AI. (2022). 16th State of Agile Report. https://info.digital.ai/rs/981-LQX-968/images/AR-SA-2022-16th-Annual-State-Of-Agile-Report.pdf
12. Estefan, J. A. (2008). Survey of Model-Based Systems Engineering (MBSE) Methodologies. INCOSE MBSE Initiative.
13. Friedenthal, S., Moore, A., & Steiner, R. (2015). A Practical Guide to SysML: The Systems Modeling Language (3rd ed.). Morgan Kaufmann.
14. Ghezzi, A., & Cavallo, A. (2020). Agile business model innovation in digital entrepreneurship: Lean startup approaches. Journal of Business Research, 110, 519–537.
15. Hadar, I., Sherman, S., Hadar, E., & Harrison, J. J. (2013). Less is more: Architecture documentation for agile development. In International Workshop on Cooperative and Human Aspects of Software Engineering.
16. Holmes, J., et al. (2022). System safety and cybersecurity within MBSE. https://doi.org/10.4271/2024-01-3966
17. Hooshmand, Y., Adamenko, D., Kunnen, S., & Köhler, P. (2017). An approach for holistic model-based engineering of industrial plants. In Proceedings of the International Conference on Engineering Design, 101–110.
18. Irnich, L., et al. (2023). Combining and evaluating MBSE solutions. https://doi.org/10.1007/s10010-023-00619-0
19. Kim, S. Y., Wagner, D., & Jimenez, A. (2019). Challenges in applying model-based systems engineering: Human-centered design perspective. INCOSE Human-Systems Integration Conference.
20. Kossiakoff, A., Biemer, S. M., Seymour, S. J., & Flanigan, D. A. (2020). Systems Engineering Principles and Practice. Wiley.
21. Kuchta, D. (2019). Combination of the earned value method and the Agile approach—A case study of a production system implementation. In Intelligent Systems in Production Engineering and Maintenance (pp. 87–96).
22. Lethbridge, T., Singer, J., & Forward, A. (2003). How software engineers use documentation: The state of the practice. IEEE Software, 20, 35–39.
23. Manifesto for Agile Software Development. (2001). http://agilemanifesto.org
24. Müller, et al. (2023). MBSE methodology and user acceptance in industry. https://doi.org/10.1016/j.procir.2023.03.135
25. Noguchi, R. A. (2019). A roadmap for advancing MBSE practice in government acquisition. INCOSE International Symposium.
26. Rinker, F., et al. (2023). Traceable multi-view model integration for agile systems engineering. https://doi.org/10.1007/s42979-022-01572-5
27. Ruparelia, N. B. (2010). Software development lifecycle models. ACM SIGSOFT Software Engineering Notes, 35, 8–13.
28. Satpathy, T. (2016). A Guide to the Scrum Body of Knowledge (SBOK™ Guide) (3rd ed.). SCRUMstudy.
29. Selic, B. (2009). Agile documentation, anyone? IEEE Software, 26, 11–12.
30. Stettina, C. J., & Heijstek, W. (2011). Necessary and neglected? Internal documentation in agile teams. In ACM Conference on Design of Communication.
31. Strobbe, C., et al. (2023). Model-based multi-disciplinary design for cyber-physical systems. https://doi.org/10.1177/10920617241295885
32. Turner, R. (2007). Toward Agile systems engineering processes. Crosstalk Journal of Defense Software Engineering, 20, 11–15.
33. U.S. Government Accountability Office. (2015). Agile Assessment Guide: Best Practices for Agile Adoption and Implementation (GAO-20-590G).
34. Van Acker, B., et al. (2023). Validity frame-enabled model-based engineering processes. https://doi.org/10.1177/00375497231205035
35. Vijayasarathy, L. R., & Turk, D. (2008). Agile software development: A survey of early adopters. Journal of Information Technology Management, XIX, 1–8.

36. Walden, D. D., Roedler, G. J., & Forsberg, K. (2015). INCOSE systems engineering handbook version 4. INCOSE International Symposium.
37. Wang, L., Izygon, M., Okon, S., Wagner, H., & Garner, L. (2016). Effort to accelerate MBSE adoption and usage at JSC. AIAA SPACE.
38. Xie, et al. (2024). X-SEM: Modeling and simulation-based systems engineering methodology. https://doi.org/10.1016/j.jmsy.2024.01.013
39. Yeo, K. T. (2002). Critical failure factors in information system projects. International Journal of Project Management, 20, 241–246.
40. Young, K. G. (2015). Defense space application of MBSE—closing the culture chasms. AIAA SPACE Conference.
41. Zimmerman, P. (2014). A review of model-based systems engineering practices and recommendations for future directions in the Department of Defense.

FUNDING

The authors received no funding for the development of this research.

CONFLICT OF INTEREST

The authors declare that there are no conflicts of interest in this study and that the processes established by this journal have been ethically followed. Furthermore, they confirm that this work has not been published, either partially or in full, in any other journal.

AUTHORSHIP CONTRIBUTIONS

Conceptualization: KDRM
Data curation: KDRM
Formal analysis: KDRM
Investigation: KDRM
Methodology: KDRM
Project administration: KDRM
Resources: KDRM
Software: KDRM
Supervision: KDRM
Validation: KDRM
Visualization: KDRM
Writing – original draft: KDRM
Writing – review and editing: KDRM