Paper ID: 244

A Comprehensive Study on Software Evolution in Plan Driven and Agile Methodologies

NS Wisidagama[1#], ML Karunarathne[2], PDCJ Paranagama[3] RMDKN Rathnayake[4], and BNS Lankasena[5]

[1,2,3,4]Department of Computer Science, General Sir Jihn Kotelawala Defence University, Ratmalana, Sri Lanka
[5]Department of Technology, University of Sri Jayawardenapura,, Sri Lanka
[#] 37-se-0012@kdu.ac.lk

Abstract— *The research paper examines the evolution of software development approaches, focusing specifically on thecomparison between agile and plan-driven methodologies. The paper provides an overview of the historical development of software development approaches, highlighting the shift towards more flexible and collaborative methods with the emergence of agile methodologies. The paper then explores the benefits and challenges of agile and plan-driven approaches, drawing on a range of case studies and empirical research. The research highlights the importance of effective project management, communication, and collaboration in facilitating the successful implementation of both agile and plan- driven methodologies. The paper also emphasizes the need to consider the specific needs and requirements of each software development project when deciding which approach to use. Overall, the research provides insights into the evolution of software development approaches and offers practical recommendations for project managers and software developers seeking to optimize their software development processes.*

Keywords— **Agile, plan-driven, software evolution**

I. INTRODUCTION

Software evolution is an essential process for anysoftware development project, as it involves the continuous improvement and adaptation of the software to meet changing user needs and requirements. In today's rapidly changing business environment, it is critical for software development organizations to implement effective software evolution strategies to remain competitive and relevant.

Two popular methodologies used in software development are the plan-driven and agile methodologies. The plan-driven approach involves a structured and sequential process, where the entire software development process is planned in advance and followed rigorously. In contrast, the agile methodology is a more flexible and adaptive approach that emphasizes continuous collaboration and iteration between developers and stakeholders. Software development methodologies play a crucial role in shaping the evolution of software systems. The study employs a systematic literature review approach to collect and analyze relevant research articles, conference papers, and case studies published in the field of software engineering. Through this process, a comprehensive dataset of studies focusing on software evolution in plan-driven and agile methodologies is assembled The software evolution process in both plan-driven and agile methodologies are explored and compared directly along with

their effectiveness in organizations. The benefits and drawbacks of each approach will be explored and provide insights into how organizations can successfully implement these methodologies for effective software evolution. Additionally, the challenges faced by organizations during the software evolution process will be discussed and strategies to overcome them.

The research paper aims to examine the software evolution process in plan-driven and agile methodologies in organizations. The scope of the study includes an overview of software development methodologies, a detailed analysis of the software evolution process, a comparative analysis of the effectiveness of software evolution in plan-driven and agile methodologies, an examination of the challenges faced by organizations during the software evolution process, case studies of organizations that have successfully implemented plan-driven and agile methodologies for software evolution, and a discussion of future research directions. The rationale and justification for the study lie in the need to understand the software evolution process in plan-driven and agile methodologies and identify best practices that can help organizations improve their software development practices. The research problem is the lack of empirical evidence of the effectiveness of plan-driven and agile methodologies for software evolution. The research objectives are to examine the software evolution process in plan-driven and agile methodologies, compare the effectiveness of software evolution in plan-driven and agile methodologies, identify challenges faced by organizations during the software evolution process, and provide practical insights and recommendations for organizations to improve their software development practices. The expected outcomes of the study include practical implications, theoretical implications, advancement of knowledge, and better decision-making. The overall goal of this research paper is to provide a comprehensive understanding of software evolution in plan- driven and agile methodologies and help organizations make informed decisions about which approach is best suited for their software development projects. Ultimately, by understanding the software evolution process, organizations can ensure the continued success and relevance of their software products in the ever-changing business environment. Some of the organizations which uses agile, whereas some uses plan-driven and also another set of companies uses both agile and plan-driven methodologies for the development.

II. LITERATURE REVIEW

(Sindhgatta, Narendra and Sengupta, 2018) The paper examines the use of agile methodologies in a software development project at a multinational IT services company. The study found that the use of agile methodologies facilitated quick and effective responses to changes in requirements and customer feedback, resulting in a higher- quality product. Additionally, agile methodologies promoted collaboration and communication between team members, leading to better coordination and improved projectperformance. However, the study also identified challenges associated with agile methodologies, such as the need for careful planning, communication, and continuous monitoring and evaluation to ensure effective application. Overall, thestudy concludes that while agile methodologies have their challenges, they can be highly effective in software development projects that involve changing requirements or a high degree of uncertainty.

(Louis, 2005) discusses the benefits and challenges of transitioning from a plan-driven culture to an agile development culture. Plan- driven approaches can be rigid and inflexible, leading to delays, cost overruns, and poor-quality outcomes. In contrast, agile development emphasizes flexibility, adaptability, and collaboration, resulting in improved outcomes and higher customer satisfaction. However, transitioning to an agile culture can be challenging due to resistance to change, lack of understanding of agile principles, and the need for new skills and roles. The author outlines a roadmap for making the transition, including developing a shared understanding of agile principles, providing training and support for team members, and creating a culture of continuous improvement. Overall, the paper emphasizes the importance of strong leadership, communication, and ongoing monitoring and evaluation in facilitating the transition to an agile culture.

(Petersen and Wohlin, 2010) The case study examines the impact of adopting agile practices in a large, multinational software development organization. The study found that the use of agile practices led to improved communication, collaboration, and flexibility within the development team, resulting in higher-quality products and increased customer satisfaction. Additionally, agile practices allowed for better management of changing requirements and reduced the risk of project failure. However, the study also identified challenges associated with the transition, including the need for cultural change, new skills and roles, and effective training and support for team members. Overall, the study demonstrates the potentialbenefits and challenges of transitioning from a plan- driven to an incremental software development approach with agile practices in a large organization. Accordingly the agile practices are most suitable for more flexible projects with changing user requirements, so that it can change according to the requirements of the users which evolves as the evolving world and needs.The study highlights the importance of careful planning, effective communication, and a commitment to cultural change within the organization to ensure the successful adoption of agile practices.

(Heeager and Nielsen, 2020) Lise Tordrup Heeager and Peter Axel Nielsen discuss the integration of agile and plan-driven development in safety- critical software development. The authors suggest that a hybrid approach that combines the flexibility and responsiveness of agile with the structure and predictability of plan-driven development can be beneficial for safety-critical software development projects. The study presents a case study of integrating agile and plan-driven development in a safety-critical software development project, highlighting the benefits and challenges of such an approach. The integrationof agile and plan-driven development allowed for improved communication and collaboration between development teams and stakeholders, resulting in a better understanding of project goals and requirements. However, the study also identified challenges associated with integrating agile and plan-driven development, including the need for effective coordination between development teams and the potential for conflictsbetween the two approaches. The authors conclude that a hybrid approach of integrating agile and plan- driven development can be effective for safety-critical software development projects, but it requires careful planning, coordination, and communication to ensure the success of the approach. The study emphasizes the importance of considering the specific needs and requirements of each software development project when deciding which approach to use.

(Laux and Kranz, 2019) Authors examine the coexistence of plan-driven and agile methods in software development projects. The study investigates how tensions can arise between the two approaches and how these tensions can be resolved to achieve a successful coexistence of the two methods. The authors suggest that differences in project goals, team structure, and communication can contribute to the emergence of tensions. To resolve these tensions, the study recommends adaptiveproject management, effective communication, and a shared understanding of project goals. It emphasizes the importance of complementary rather than conflicting methods for a successful coexistence of plan-driven and agile approaches. The paper highlights the need to consider the specific needs and requirements of each software development project when deciding which approach to use. Overall, the study provides insights into the coexistence of plan-driven and agile methods in software development and suggests ways to overcome tensions between the two approaches to achieve a successful and effective software development process.

(Marinho et al., 2019) The authors suggest that plan-driven approaches are still prevalent in GSD projects despite the widespread adoption of agile methods. There are several challenges and, the paper highlights the challenges associated with GSD projects, such as communication, coordination, and cultural differences, which may limit the effectiveness of strict adherence to agile methods. The study recommends a hybrid approach that combines plan-driven and agile methods for greater flexibility and adaptability while still providing a clear plan and structure. The authors emphasize the importance of understanding the specific needs and requirements of eachGSD project and tailoring the development approach accordingly. The study highlights the need for further researchto explore the effectiveness of different hybrid approaches in GSD projects. Overall, the paper provides insights into the coexistence of plan-driven and agile methods in GSD projects and suggests that a hybrid approach can be more effective in such projects. The study calls for a tailored and flexible approach that takes into account the unique challenges and requirements of each GSD project.

(Silva, Bianchi and Amaral, 2019) The authors explore the challenges of integrating agile and plan-driven project management approaches. The authors propose a framework for evaluating and selecting the most appropriate combination of methods based on project requirements, characteristics, and team skills. The paper identifies the benefits and drawbacks of both agile and plan- driven approaches and argues that a combination of these two methods can provide an optimal solution for certain projects. The authors present a case study of a software development project that successfully implemented a combined approach and achieved better project outcomes compared to previous projects that used only agile or plan- driven approaches. The paper also discusses the importance of communication, collaboration, and team training in successfully implementinga combined approach. The authors conclude that a combined approach can provide a flexible and adaptable project management solution that can improve project outcomes and meet the specific needs of each project. Overall, the paper provides a valuable contribution to the field of project management by presenting a framework for evaluating and selecting the most appropriate project management approach based on project requirements and team skills.

(Svensson, 2005) The authors examine the challenges of integrating agile and plan-driven methodologies and propose a software tool to support this integration. The paper argues that both approacheshave their strengths and weaknesses and that a hybrid approach can provide a better solution to managing software development projects. The author proposes a tool called APL (Agile and Plan-driven method Language) that provides a common language and framework for managing software development projects that use both agile and plan- driven approaches. APL enables project managers to define project requirements, manage project resources, and monitor project progress using a combination of agile and plan-driven methodologies. The paper presents a case study of a software development project that successfully implemented APL and achieved better project outcomes compared to previousprojects that used only agile or plan-driven methodologies. The paper concludes that APL can provide an effective
solution for managing software development projects that require a combination of agile and plan-driven methodologies. Overall, the paper presents a valuable contribution to the field of software development project management by providing a tool that supports the integration of agile and plan-driven methodologies.

(Khalil, 2018) The research paper explores the use of agile and plan- driven approaches in information and communication technology (ICT) development projects. The paper presents the findings of a survey conducted with 168 participants from 22 countries to understand the state of practice in the use of agile and plan-driven approaches in ICT development projects. The study reveals that both approaches are commonly used, with a higher prevalence of agile approaches.Agile approaches were preferred for their flexibility and adaptability, while plan-driven approaches were preferred for their predictability and control. The study also found that organizations often integrate the two approaches to take advantage of their strengths. However, the paper suggests that more research is needed to better understand the factors that influence the choice between the two approaches and how they can be integrated effectively. Overall, the paper contributes to the field of software development project management by providing insights into the state of practice of agile and plan-driven approaches in ICT development projects. The study highlights the need for a better understanding of the factors that influence the choice between these two approaches and the potential benefits of integrating them. The results of this study can help organizations make informed decisions about which approach to adopt based on their project needs and goals.

(Gren, Wong andKristo, 2017) The research paper analyzes the implementation of agile and plan-driven approaches in ERP projects. The paper presents the findings of a study that examined 21 ERP implementations from 20 companies across different industries to understand the factors that influence the choice between agile and plan-driven approaches. The study found that both approaches are viable for ERP implementations, but the choice between them depends on the project context and goals. Agile approaches were preferred for their flexibility and adaptability, while plan-driven approaches were preferred for their predictability and control. The study also found that organizations often use a combination of agile and plan-driven approaches to take advantage of their strengths. The paper highlights the importance of understanding the project context and goals when choosing between agile and plan-driven approaches. It also emphasizes the need for project managers to have a good understanding of both approaches and theability to adapt to changing project requirements. Overall, the paper contributes to the field of ERP project management by providing insights into the factors that influence the choice between agile and plan-driven approaches. The study's findings can help organizations make informed decisions about which approach to adopt based on their project needsand goals and can also guide project managers in selecting themost appropriate approach for their projects. (de O. Melo et al., 2013) The paper presents an industrial case study that examines the effects of moving from a plan-driven software development approach to an incremental approach with agile practices. The study found that the change in approach led to several benefits, including increased flexibility and adaptability, improved communication and collaboration among team members, and increased customer satisfaction. The study also revealed that the transition was not without challenges, including resistance to change from team members, difficulty in estimating the scope of work, and the need for additional training and coaching. The authorsrecommend that organizations carefully consider the benefits and challenges of adopting an incremental approach with agile practices and provide the necessary support and resources fora successful transition. Overall, the study highlights the potential benefits of adopting agile practices in software development and the importance of managing the transition effectively.

(Good, 2003) The paper describes a pragmatic approach to implementing agile software development methodologies in plan-driven organizations. The authors argue that plan-driven organizations can benefit from incorporating agile practices, but that a one-size-fits-all approach is not appropriate. Instead,they propose a phased approach that gradually introduces agile practices while minimizing disruption to existing processes. The paper also discusses key success factors for implementing agile in a plan-driven environment, including executivesupport, stakeholder engagement, and the availability of appropriate tools and infrastructure. The authors provide acase study of the successful implementation of this approach in a large organization. The study found that the phased approach enabled the organization to reap the benefits of agile practices, including increased team collaboration and faster time-to-market while minimizing the risk of disruption. Overall, the paper provides practical guidance for organizations looking to incorporate agile practices into their software development processes.

(Turner and Boehm,2003) The research paper highlights the importance of people factors in software management and compares the agile and plan-driven methods. The study reveals that although both approaches have their strengths and weaknesses, the successof the project ultimately depends on the people involved. Agile methods focus on communication and collaboration, which are essential for success. Agile also promotes self-organizing teams, which can lead to higher job satisfactionand motivation. On the other hand, plan-driven methods emphasize processes and documentation, which can be more suitable for projects with complex regulatory requirements. However, they can also lead to rigid and inflexible teams. The paper concludes that it is essential to consider people factors, such as motivation, communication, and teamwork, when choosing a software management approach. By doing so, organizations can increase their chances of success and better meet the needs of their stakeholders.

(Stefan Zugal Bakk.techn., 2008) The author examines the effectiveness of agile and plan- driven approaches to project planning through a controlled experiment. The study was conducted with 24 participants, divided into two groups, where each group used either the agile or plan-driven approach. The results showed that theagile approach was more effective in terms of achieving higher customer satisfaction and better quality of software, while the plan-driven approach was more efficient in terms of better adherence to schedule and cost estimation. However, the study also revealed that the effectiveness of each approach depends on the project's characteristics, team members' expertise, and stakeholder involvement. The findings suggest that both approaches have their strengths and weaknesses, and a hybrid approach may be more suitable for certain projects. The study also emphasizes the importance of considering individual and team factors, such as motivation and communication, in selecting and implementing a suitable approach for software development projects.

III. METHODOLOGY

This study aims to investigate software evolution in plan- driven versus agile methodologies by conducting a case study on two organizations. The study will employ a mixed methods approach to gather and analyze data, including both qualitative and quantitative data. The study population will consist of two organizations that have implemented either plan-driven or agile methodologies for software development. The sample will include software development teams and senior managers involved in the software development process in organizations.

A. Data Collection

Plan driven approaches and the other that uses agile approaches. The aim is to collect data on their experiences, challenges, and benefits of using these approaches. The surveys will be administered to the team members and stakeholders involved in the software development process in each organization. The surveys will be conducted online, and the participants will be given a link to access the survey. The surveys will be anonymous, and the participants' responses will be kept confidential.

B. Data Analysis

The data will be collected through a combination of semi-structured interview and survey, and research papers. A comprehensive search of relevant research papers on plan- driven and agile approaches in software development will be conducted using online academic databases such as Google Scholar, IEEE Xplore, and ACM Digital Library. The research papers selected for the review will be based on the relevance to the research question, their publication date, and the quality of the research conducted. The selected papers will be from reputable journals and conferences in the field of software engineering. The semi-structured survey interview will be conducted with two members involved in the software development process in each organization. The interviews will be conducted online through survey, depending on the availability of the participants. A questionnaire will be developed based on the research question and objectives. The survey will be sent to two companies - one that uses plan- driven approaches and the other that uses agile approaches. The aim is to collect data on their experiences, challenges, andbenefits of using these approaches. The surveys will be administered to the team members and stakeholders involved in the software development process in each organization. Thesurveys will be conducted online, and the participants will be given a link to access the survey. The surveys will be anonymous, and the participants' responses will be kept confidential.

C. Results and Discussion

The results of the data analysis will be presented in tables and graphs. The findings will be discussed in light of the research question and objectives. The similarities and differences between plan-driven and agile approaches will be highlighted. The strengths and weaknesses of each approach will be discussed, and recommendations will be made on which approach is suitable for different types of software projects.

D. Ethical Considerations

The study will adhere to ethical guidelines, and informed consent will be obtained from all participants. Participants' anonymity and confidentiality will be maintained throughout the study. Any identifiable information collected from the participants will be kept confidential and will not be shared with any third party.

IV. RESULTS

This section presents the results of the case study research conducted on two organizations, Intelligent Automation Heritage Bank and Peerless Foods Organization, in order to compare their software evolution processes using agile and plan-driven methodologies, respectively. The data was collected through interviews with the organizations' software development teams and senior managers.

Based on the responses provided by Don Kannangarafrom Intelligent Automation Heritage Bank, and the representative of Peerless Foods Organization, the following results can be highlighted:

A. Quality measurement

Both companies use different methodologies to measure the quality of their software. Heritage Bank uses Agile methodology and measures quality based on how well the solution meets business needs and requirements, how well sprints and iterations executed to deliver software on time and within budget, and how well continuous integration and deployment is utilized to catch defects and ensure that software is always in a releasable state. On the other hand, Peerless Foods Organization uses Plan-driven methodology and measures quality based on a comprehensive set of quality metrics, such as defect density, code coverage, and test coverage, which they track to ensure that the software meets project objectives. They perform various types of testing, use automated testing tools to increase efficiency and effectiveness, conduct peer reviews of code, design, and requirements, and actively seek feedback from customers through surveys and focus groups to continuously improve thesoftware.

B. *Quality of software produced using Agile vs Plan-driven methodologies*

Peerless Foods Organization did not compare the quality of software produced using Agile vs Plan-driven methodologies. On the other hand, Don Kannangara from Intelligent Automation Heritage Bank stated that quality does not necessarily depend on the methodology but how well the chosen methodology is applied, how good and experienced the project team is, how well the requirements have been captured and tested, as well as how the overall project is planned and executed.

C. *Maintenance using Agile vs Plan-driven methodologies Heritage Bank approaches*

software maintenance by collaborating closely with business stakeholders, maintaining a backlog of work, assessing the backlog and prioritizing based on ROI, delivery time, criticality and urgency, using time- boxed Sprints to deliver the backlog, and using retrospective meetings to improve delivery processes. They ensure that software maintenance is performed efficiently and effectively by collaborating closely with business stakeholders to understand their needs and priorities, building and maintaining a plan, applying a clear and consistent work prioritization approach, using data and analytics to assist with the prioritization, using agile methodologies to deliver regularly and efficiently, using automated testing, and enforcing clear Change management, risk assessment and software delivery best practices consistently. On the other hand, Peerless Foods Organization approaches software maintenance by identifying the software components that require maintenance, prioritizing the maintenance tasks, developing a maintenance plan, executing the plan, and documenting the maintenance activities. Their approach emphasizes structured processes, thorough documentation, anda focus on quality to ensure the software continues to function correctly and meet customer needs. They ensure that software maintenance is performed efficiently and effectively by establishing a well-defined process for maintenance, using automated tools to support the process, and ensuring the maintenance team has the necessary skills and knowledge.

D. Challenges

Both companies faced challenges associated with their methodologies. Heritage Bank faced challenges in changing the mindset and culture of people who have not been used to Agile delivery methodologies, inexperienced team members who struggle to understand the principles and practices, and unclear requirements due to not iterating enough and not receiving/seeking feedback from the stakeholders. On the other hand, Peerless Foods Organization faced challenges such as limited flexibility, complex documentation, risk management, overemphasis on planning, and limited stakeholder engagement. These challenges led to delays, increased costs, and difficulty delivering high-quality software on time and within budget.

E. *Improvements*

To address the challenges, Heritage Bank continuously trains and mentors junior colleagues, uses Agile project management software, automated testing tools, and collaboration platforms, such as JIRA and Kanban, uses retrospective meetings to improve delivery skills and outcomes, collaborates closely with business stakeholders and seeks their regular feedback, and picks the methodology basedon the size and type of the project and team size. On the other hand, Peerless Foods Organization has addressed the challenges.

Based on the responses provided by the two interviewees, it appears that both companies place a significant emphasis on ensuring the quality of their software products. However, they approach this goal in slightly different ways due to their choice of methodology.

The Heritage Bank uses Agile methodology and measuresthe quality of their software by evaluating how well it meets business needs and requirements, how well sprints and iterations are executed, and how well continuous integration and deployment are utilized. The company also uses code and peer reviews, thorough testing (including automated test scenarios), and continuous performance monitoring and analytics to maintain quality throughout the software evolution process. In terms of software maintenance, Heritage Bank collaborates closely with business stakeholders, maintains a backlog of work, and uses time-boxed sprints to deliver the backlog, while using retrospectives meetings to improve delivery processes.

On the other hand, Peerless Foods uses a Plan-driven methodology and measures software quality through a comprehensive set of quality metrics, such as defect density, code coverage, and test coverage. The company performs various types of testing, uses automated testing tools, conducts peer reviews of code, design, and requirements, and uses code analysis tools to identify potential issues early in the development process. Peerless Foods also actively seeks feedback from customers through surveys and focus groups to continuously improve the software. In terms of software maintenance, the company identifies the software components that require maintenance, prioritizes the maintenance tasks, develops a maintenance plan, executes the plan, and documents the maintenance activities.

It's worth noting that neither company reported any significant differences in the quality of software produced using Agile versus Plan-driven methodologies. This suggests that while the methodology can impact how quality is maintained, the ultimate success depends on how well the methodology is implemented and how closely the team adheres to best practices.

Both companies also faced unique challenges associated with their chosen methodology. Heritage Bank had to overcome the challenges of changing the mindset and culture of people who were not used to Agile delivery methodologies, inexperienced team members struggling to understand the principles and practices, and unclear requirements due to insufficient iterations and feedback. Peerless Foods faced challenges such as limited flexibility, complex documentation,overemphasis on planning, and limited stakeholder engagement.

To address these challenges, Heritage Bank continuously trained their team members and used Agile project management software, automated testing tools, and collaboration platforms such as JIRA and Kanban. The company also collaborated closely with business stakeholders and regularly sought their feedback, while selecting the methodology that best suited the project in hand. Peerless Foods addressed the challenges associated with Plan-driven methodology by implementing a change control process, developing streamlined templates and tools for documentation,establishing a risk management team and process, implementing agile principles, and establishing regular communication channels with stakeholders.

In conclusion, both Heritage Bank and Peerless Foodsprioritize maintaining software quality through a combination of testing, reviews, change management, and customer feedback, albeit using different methodologies. While eachmethodology has its own challenges, both companies have taken steps to address these challenges and continuously improve their processes over time.

V. DISCUSSION

Software evolution is an essential aspect of software development that encompasses the maintenance, modification, and evolution of software systems. Software evolution is a continuous process that requires a well-defined and structured approach to ensure that the software continues to function correctly and meet customer needs. Through this paper, we researched software evolution in Agile vs Plan-driven methodologies and how companies used them in practice. From studying the existing research and theories , analyzing how organizations use the two methodologies and how these help for the betterment and future success of the organization we could see that agile methodology emphasizes collaboration, customer involvement, and flexibility. The success of agile is measured by how well the solution meets business needs and requirements, how well sprints and iterations are executed, and how well continuous integration and deployment are utilized. Agile methodology requires a highly experienced project team that can collaborate closely with business stakeholders, maintain a backlog of work, and use time-boxed sprints to deliver the backlog. The Agile methodology also involves thorough testing, including automated test scenarios for wider coverage, and continuous performance monitoring and analytics. However, Agile methodology may face challenges in changing the mindset and culture of people who have not been used to Agile delivery methodologies, lack of experience in Agile delivery, and unclear requirements due to not iterating enough and not receiving/seeking feedback from the stakeholders. In contrast, Plan-driven methodology emphasizes structured processes, thorough documentation, and a focus on quality to ensure the software continues to function correctly and meet customer needs. Plan-driven methodology uses comprehensive quality metrics, conducts several types of testing, and actively seeks feedback from customers to continuously improve the software. Plan-driven methodology has limitations such as limited flexibility, complex documentation, risk management, overemphasis on planning, and limited stakeholder engagement. However, Plan-driven methodology has addressed these challenges by implementing a change control process, developing streamlined templates and tools for documentation, establishing a risk management team and process, implementing Agile principles, and establishing regular communication channels with stakeholders. In conclusion, software evolution in Agile vs Plan-driven methodologies requires a well-defined and structuredapproach to ensure that the software continues to functioncorrectly and meet customer needs. Agile methodology requires a highly experienced project team that can collaborate closely with business stakeholders, while Plan-driven methodology emphasizes structured processes, thorough documentation, and a focus on quality.

Table 1. Comparison of Agile and plan-driven methodologies

	Agile	**Plan-driven**
Approach	Iterative incremental	Sequential linear
Process	Adaptive flexible	Rigid inflexible
Requirements	Prioritizes customer feedback and changing requirements	Emphasizes detailed upfro nt planning a nd documentation
Communicatio n	Collaborative frequent	Formal occasional
Documentation	Minimalist informal	Detailed formal
Risk manage ment	Continuous identification mitigation	Predefined ri sk management plan
Team structure	Self-organizing and cross-functional teams	Hierarchical a nd specialized teams

This table highlights some of the key differences between agile and plan-driven approaches in software development. Agile approaches prioritize adaptability and customer feedback, while plan-driven approaches emphasize upfront planning and documentation. Communication and team structures also differ between the two approaches, with agile approaches promoting collaboration and self-organizing teams and plan-driven approaches relying on formal communication and specialized team roles. Ultimately, the choice between agile and plan-driven approaches depends on the specific needs and requirements of each software development project.

VI. CONCLUSION

After conducting research on software evolution in plan-driven versus agile methodologies and interviewing two companies that use these methodologies, it is clear that both approaches have their advantages and disadvantages.

Plan-driven methodologies provide a structured approach to software development, which can be beneficial for large projects with well-defined requirements. However, they may not be flexible enough to handle changes that arise during the development process. On the other hand, agile methodologies prioritize flexibility and adaptability, making them a good fit for projects with evolving requirements. However, they can lead to scope creep and may not provide enough structure for large, complex projects. The interviews with the two companies revealed that the choice between plan-driven and agile methodologies depends on various factors, such as project scope, team size, and project requirements. Both companies had positive experiences with their chosen methodologies but also encountered some challenges. Overall, the research suggests that there is no one-size-fits-all approach to software development. Project managers and teams must carefully consider their specific needs and choose the methodology that best suits their situation. Additionally, it is important to remain flexible and willing to adapt as the projectevolves to ensure the best possible outcome.

REFERENCES

Akbar, R.I. (2013) ‘COMPARISON BETWEEN AGILE METHODOLOGY AND PLAN DRIVEN AS METHOD FOR ERP DEVELOPMENT’.

Good, J.M. (2003) ‘A Pragmatic Approach to the Implementation of Agile Software Development Methodologies in Plan-Driven Organisations’.

Gren, L., Wong, A. and Kristo, E. (2017) ‘Choosing agile or plan-driven enterprise resource planning (ERP) implementations— A study on 21 implementations from 20 companies’.

Hadar, I. and Sherman, S. (2012) ‘Agile vs. plan-driven perceptions of software architecture’, in 2012 5th International Workshop on Co-operative and Human Aspects of Software Engineering (CHASE). 2012 5th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), Zurich, Switzerland: IEEE, pp. 50–55. Available at: https://doi.org/10.1109/CHASE.2012.6223022.

Heeager, L.T. and Nielsen, P.A. (2020) ‘Meshing agile and plan- driven development in safety-critical software: a case study’, Empirical Software Engineering, 25(2), pp. 1035– 1062. Available at: https://doi.org/10.1007/s10664-020- 09804-z.

Jalali, S. and Wohlin, C. (2012) ‘Global software engineering and agile practices: a systematic review: GLOBAL SOFTWARE ENGINEERING AND AGILE PRACTICES’,

Journal of Software: Evolution and Process, 24(6), pp. 643– 659. Available at: https://doi.org/10.1002/smr.561.

Khalil, C. (2018) ‘The State of the Practice of Agile and Plan-Driven Approaches in ICT Development Projects: An Exploratory Research Study’, in C. Rossignoli, F. Virili, and

S. Za (eds) Digital Technology and Organizational Change. Cham: Springer International Publishing (Lecture Notes in Information Systems and Organisation), pp. 25–33. Available at: https://doi.org/10.1007/978-3-319-62051-0_3.

Laux, I. and Kranz, J. (2019) ‘Coexisting Plan-driven andAgile Methods: How Tensions Emerge and Are Resolved’.

Louis, S. (2005) ‘Moving from a Plan Driven Culture to Agile Development’.

Marinho, M. et al. (2019) ‘Plan-Driven Approaches Are Alive and Kicking in Agile Global Software Development’, in 2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). 2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), Porto de Galinhas, Recife, Brazil: IEEE,pp. 1–11. Available at: https://doi.org/10.1109/ESEM.2019.8870168.

Menon, V., Sinha, R. and MacDonell, S. (2015) ‘Architectural Challenges in Migrating Plan-driven Projects to Agile’:, in Proceedings of the 10th International Conference on Evaluation of Novel Approaches to Software Engineering. 10th International Conference on Evaluation of Novel Software Approaches to Software Engineering, Barcelona, Spain: SCITEPRESS - Science and and Technology Publications, pp. 223–228. Available at: https://doi.org/10.5220/0005383502230228.

de O. Melo, C. et al. (2013) ‘The evolution of agile software development in Brazil: Education, research, and the state-of-the-practice’, Journal of the Brazilian Computer Society, 19(4),pp. 523–552. Available at:https://doi.org/10.1007/s13173-013-0114-x.

Petersen, K. and Wohlin, C. (2010) ‘The effect of moving from a plan-driven to an incremental software development approach with agile practices: An industrial case study’, Empirical Software Engineering, 15(6), pp. 654–693. Available at: https://doi.org/10.1007/s10664-010-9136-6.

Silva, F.B., Bianchi, M.J. and Amaral, D.C. (2019) ‘Evaluating combined project management models: strategies for agile and plan-driven integration’, Product Management & Development, 17(1), pp. 15–30. Available at: https://doi.org/10.4322/pmd.2019.003.

Sindhgatta, R., Narendra, N.C. and Sengupta, B. (no date) ‘Software evolution in agile development: a case study’.

Stefan Zugal Bakk.techn. (2008) ‘Agile versus Plan-Driven Approaches to Planning-A Controlled Experiment’.

Svensson, H. (2005) ‘Developing Support for Agile and Plan-Driven Methods’.

Turner, R. and Boehm, B. (2003) ‘People Factors in Software Management: Lessons From Comparing Agile and Plan- Driven Methods’.

Waseem, M. and Ikram, N. (2016) ‘Architecting Activities Evolution and Emergence in Agile Software Development: An Empirical Investigation: Initial Research Proposal’, in H. Sharp and T. Hall (eds) Agile Processes, in SoftwareEngineering, and Extreme Programming. Cham: Springer International Publishing (Lecture Notes in Business Information Processing), pp. 326–332. Available at: https://doi.org/10.1007/978-3-319-33515-5_35.

Wlodarski, R., Poniszewska-Maranda, A. and Falleri, J.-R. (2020) ‘Comparative Case Study of Plan-Driven and Agile Approaches in Student Computing Projects’, in 2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM). 2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), Split, Croatia: IEEE, pp. 1–6. Available at: https://doi.org/10.23919/SoftCOM50211.2020.9238196.