Journal of Statistics & Management Systems
ISSN 0972-0510 (Print), ISSN 2169-0014 (Online)
Vol. 28 (2025), No. 6, pp. 1167–1178
DOI : 10.47974/JSMS-1482

Analyzing the impact of agile practices on software projects

Mukesh Kumar *
Arvind Kalia †
Department of Computer Science
Himachal Pradesh University
Shimla 171005
Himachal Pradesh
India

Pardeep Kumar Mittal §
Department of Computer Science and Applications
Kurukshetra University
Kurukshetra 136119
Haryana
India

Naveen Kumar ‡
Department of Computer Science
Himachal Pradesh University
Shimla 171005
Himachal Pradesh
India

Abstract

The effectiveness of a software project can be defined differently by different people. For writing purposes, it is introducing a software product to the market to deliver the conceived users within some pre-agreed timetable and within budget that is acceptable to the supporter of the venture. This definition of success is significantly caused by the notorious issue of development time slippage and software cost escalation of the products. These factors were particularly challenging, as deal with when software development adopted the conventional, waterfall-style forms that historically had low success rates.

As the years have gone by, software project development practices have been able to diminish these catastrophes. With agile practices, it becomes a separate ball game altogether.

* *E-mail:* **mukeshyadav1708144@gmail.com (Corresponding Author)**
† *E-mail:* **arvkalia@gmail.com**
§ *E-mail:* **pkmittal@kuk.ac.in**
‡ *E-mail:* **nkjaglan@gmail.com**

Agile methodologies have an underlying foundation of methods, ideas, and values. Though the single agile systems may deviate marginally, all have certain primary practices that emanate from four overarching values and twelve guiding philosophies. These values complement the purpose of software development as a whole and the single purpose, like focusing on the production of software that works rather than devoting so much time on documentation, but the ideas can be contradicting at times, like the challenge of achieving high quality with little cost.

Subject Classification: *68N30.*

Keywords: *Agile project development, Agile forest, Software parameters.*

1. Introduction

Software development through Agile offers an adjunct flexible process of generating the plan-driven methods of old. With that flexibility comes the fact that it can be adapted to accommodate changing requirements whatever the phase of the SDLC [1]. Its emphasis on association between the clients and the developers and supporting the developers as members of the team who can be self-organizing [2], agile methodology emphasizes maintaining the development process' spirit [3]. A number of these agile practices can be utilized on projects to do more of the above [4, 5]. Little has been researched on the way that a suite of different agile methods and practices has an influence on the duration and cost of the achievement of projects. Such emphasis necessitates more research on how different agile methods impact the outcome in a project. The research in this case tracks the implemented practices in agile software projects and examines how their collection impacts overall success in a project. The paper sets out by discussing existing research on agile methodologies and practices and proceeds to examine how the combination of different practices impacts a project outcome. The paper outlines thus: Part 2 provides a review on corresponding research and a summary on existing work; Part 3 provides the study design; Part 4 provides methodology and procedures followed; Section 5, the results are analysed and discussed, whereas Section 6 concludes the study and proposes potential areas for further investigation.

2. Literature Review

Towards fully comprehending the space of this study, it's important to consider the history of project development. The subject took form

during the 1950s and 1960s due to the necessity to enhance efficiency through improved tools and development practices. It was formally recognized as a dedicated discipline during the NATO conference of 1968. Subsequent decades witnessed progress through structured programming, CASE tools, UML, and object-oriented programming, each of which helped alleviate early obstacles in project development. Agile methodologies later emerged as a response to two persistent challenges: the need for stronger collaboration within development teams to achieve quality outcomes, and active client involvement to keep software adaptable as requirements evolve [6-9].

Although the SDLC offers numerous benefits, its adoption may impose significant financial burdens on emerging businesses and SMEs, owing to ongoing expenses like system upkeep, hardware enhancements, and staff training. Over time, project development has undergone significant shifts, with agile frameworks proving central to managing modern demands. Proponents argue that team members should be valued as key contributors rather than replaceable components, a philosophy that enhances product quality while reducing delays and financial overruns [10-12].

Researchers have developed matrices to assist in choosing appropriate approaches in IT projects, particularly to manage ongoing challenges like rising costs. Firms that adapt to technological advancements generally perform more effectively. Given the inherent complexity of software development, large-scale challenges must often be decomposed into smaller, more manageable parts. Development teams, therefore, play a critical role in evaluating which technical practices enhance success and which may hinder it [13-15].

3. Design of Study

Software development experts carefully select appropriate techniques for projects because the effectiveness of the chosen approach directly influences the outcome. Although certain methodologies often generate enthusiasm, in reality, teams rarely follow a single method in a rigid manner. The key factor is that delivering functional systems takes priority, whereas several approaches focus heavily on planning and documentation. Effectiveness is typically evaluated based on whether the system achieves its intended goals, even if the process diverges from the original design [16-18]. These highpoints a preference for practical achievements rather

than strict compliance with predefined methods, suggesting that any methodology adopted should allow for adaptability and flexibility [19].

When tailoring an agile approach to project execution, the focus should not be restricted to one predefined framework. Instead, attention must remain on the quality and usability of the final software. A more beneficial approach involves selecting agile practices that best suit each development cycle or increment, rather than fixing all practices in advance for the entire project [18]. This strategy requires evaluating the specific requirements and circumstances of each increment at its outset and then adopting practices that align most closely with that stage.

The examination of prior research enabled the compilation of a set of agile practices that are most frequently implemented, as presented in the below given **Table 1** [3,8–10]. To verify the reliability and practical applicability of this compilation, it was evaluated by five IT experts, each possessing over eight years of experience in agile projects. Following this validation, the finalized list was shared with survey respondents, who were asked to indicate which practices they had discontinued in their own software development activities [16–17].

Table 1

Practices of Agile projects.

Agile Practice	Shorts Notes on Practices
Daily Team Huddle	A brief team meeting, usually lasting around 15 minutes, where members quickly discuss progress, upcoming tasks, and any blockers.
Integration of Code Updates	The process of integrating individual developers' code updates into a common source repository.
Prioritized Task List	An ordered list of project requirements or deliverables that guides the team on what to work on next.
Pair Programming	A technique where two programmers work together at the same system, sharing ideas and responsibilities.
Task Completion Visuals	Graphical representations used to track completed tasks against remaining work.
Definition of Task Completion	A predefined set of conditions that must be met before a deliverable is accepted as complete.
Code Refinement and Enhancement	Refining the structure and efficiency of code without changing its external behaviour.
Sprint Progress Board	A visual board used to monitor the status of different tasks during a sprint.

Visual Task Flow Management	A tool that displays team tasks and processes in a visual manner to enhance coordination.
Retrospective Session	A discussion session where the team reviews achievements, challenges, and insights gained.
User-Centered Work Narratives	Concise narratives that capture specific goals or needs from the perspective of a user.
Development Timebox	A fixed-length development phase during which a selected set of tasks is completed.
Software Capability Descriptions	Descriptions of what the software is expected to do, focusing on user-oriented outcomes.
Workload Assessment	A collaborative practice where team members assess the time or effort required for tasks.
Target User Descriptions	Detailed characterizations of target or potential users, outlining their goals and behaviors.
Automated Test Execution	Automated tests that run predefined scenarios whenever changes are introduced in the code.
Software for Project Monitoring	Applications or platforms employed to track progress, share updates, and manage the project.
Task Preparedness Guide	A set of guidelines used to verify whether a task is sufficiently prepared to be started.
Module-Level Validation	The evaluation of individual software modules to ensure they function as expected.
Continuous Code Incorporation	The practice of frequently combining code changes into the central project to maintain consistency.

4. Collection of Data

4.1. *Technique*

The data collection was carried out through an online survey platform. Invitations were distributed via email to software developers involved in various project activities across India, using both individualized participant links and general organizational links. Completing the questionnaire required approximately ten minutes on average. Over a span of four months, the survey yielded a total of 73 valid responses.

4.2. *Population*

The survey yielded 68 usable responses from professionals affiliated with fifty distinct organizations. To gain deeper insights into the respondent profile, details regarding organizational size (Table 2) and the

primary nature of their software development activities (Table 3) were examined. The largest proportion of participants (34%) were employed in medium-sized organizations with 61–299 employees. Conversely, fourteen respondents (16.2%) reported working in very small firms with fewer than five employees.

With respect to development orientation, half of the respondents (36 individuals) indicated that their organizations were primarily engaged in product development. Another 35% (26 respondents) reported involvement in in-house development, while 12% (09 respondents) were associated within tailored solutions providers (Table 3)

Survey respondents indicated their professional designations within their organizations (Table 4). Most participants were developers (40 respondents, 55.6%), whereas only a single individual (1%) identified as an Integrator

Table 2
Organizational Profile

Employees in Numbers	No. of Respondents	Average
Upwards 1800	3	6.7
1100–1800	5	9.3
300–1099	4	4.2
61–299	23	34.5
30–60	11	15.3
11–29	3	4.5
5–10	5	9.4
Below 4	14	16.2

Table 3
Business Classification

Category of Organization	Number of Respondents	Percentage (%)
Product-oriented firms	36	49
In-house solutions providers	26	35
Custom service companies	9	12
Miscellaneous/Other	2	4

Table 4
Roles of Participants

Designation	Number of Respondents	Percentage (%)
Software Developer (SD)	40	55.6
Software Tester (ST)	23	31.9
Scrum Master (SM)	14	19.4
Business Analyst (BA)	13	18.1
Team Lead (TL)	12	16.7
Product Owner (PO)	10	13.9
Project Manager (PM)	8	11.1
Line Manager (LM)	7	9.7
Technical Trainer	5	6.9
Code Reviewer (CR)	5	6.9
Agile Mentor (AM)	5	6.9
Technical Writer (TW)	4	5.6
Other Roles	3	4.2
UI/UX Specialist	2	2.8
Academic/Industry Guide	2	2.8
System Integrator	1	1.4

The distribution of respondents' professional experience reveals considerable diversity [19]. The largest segment (28.6%) consisted of highly experienced professionals with more than 20 years in the field, highlighting strong representation from senior practitioners. Mid-level experience groups were also well represented, with 15.6% having between 6 and 8 years and 11.7% reporting 9–11 years of experience. At the early career stage, 11.6% of participants had less than three years of professional involvement, while another 10.4% fell within the 3–5 years range. The varied experience levels among respondents suggest that the survey captured insights from both veteran and early-career software professionals, contributing to the robustness of the findings. Relevant information is summarized in the table 5. [12–15]

Table 5

Respondent experiences [19].

Experience (in Years)	Number of Respondents	Percentage (%)
20 years and above	22	28.6
15 – 19 years	10	13.0
12 – 14 years	7	9.1
9 – 11 years	9	11.7
6 – 8 years	12	15.6
3 – 5 years	8	10.4
Less than 3 years	9	11.6

Figure 1 provides an overview of the agile practices adopted across development projects. On average, each team implemented approximately eleven practices (M = 11), reflecting the breadth of techniques incorporated into their development activities.

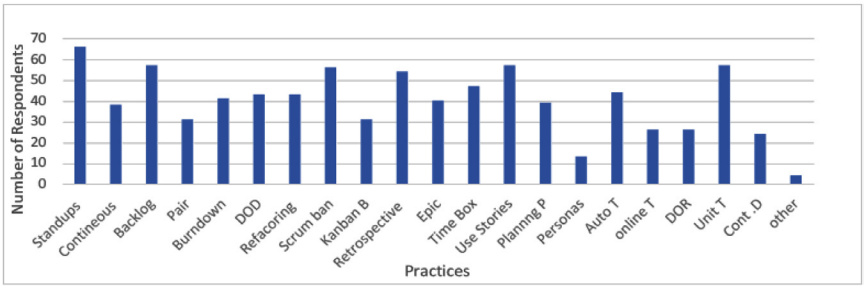

Figure 1

Projects Practices

5. Outcomes

5.1 *Team's Challenges*

Issues arising within software development teams can have a major impact on project outcomes. To explore this, the survey included questions on collaboration difficulties. Regarding communication problems during a sprint, 38 respondents (52%) reported encountering them one to three times, whereas 17 respondents (19.2%) stated they did not face any. A full breakdown is presented in Table 6.

Table 6

Challenges in a Team.

Frequency of Occurrence	0 Times	1-3 Times	5-7 Times	8 -9 Times	> 9 Times	Overall
Obstacles to collaboration	13	37	14	3	2	74
Personal challenges	19	41	7	3	3	74
Problems in sharing of ideas	21	35	9	3	2	74
Difficulties with distribution	22	32	15	1	6	74

5.2. *Project Completion*

The survey also collected information on whether projects were completed within the expected schedule and budget (Tables 7 and 8). Only one respondent (1.4%) finished earlier than planned, while 39 participants (53.4%) delivered on time. Meanwhile, 18 respondents (24.7%) experienced delays beyond the scheduled timeframe.

Table 7

Project accomplishment time.

Status	Respondents Count	%
Before Time	2	13.70%
Exact on time	40	56.16%
Late	20	28.56%
Can't Say	11	1.37%

Table 8

Project Cost

Financial Performance	Respondents Count	Avg.
Not Analysed	28	39.6
Predefined Cost	23	33.2
Beyond Cost	16	23.0
Below cost	2	2.7

With respect to project budgets, 28 respondents (39.6%) indicated that their projects were completed within the allocated financial resources, while 16 participants (23.0%) reported exceeding the planned budget.

Interestingly, the largest segment of respondents, also 28 individuals (39.6%), indicated unawareness of the project budget, highlighting a potential deficiency in financial communication or transparency among team members.

6. Conclusions

This research is based on insights from professionals engaged in Agile software development teams, highlighting both their hands-on experiences and the methods they implement. The study aims to help software development organizations in India better understand how Agile practices can address challenges related to project schedules and budget management. Practices found to have a significant impact on team communication, requirements handling, and task prioritization include Daily Stand-ups, Backlog Management, Unit Testing, User Stories, Scrumban, and Retrospective Meetings.

The findings suggest that partial or superficial adoption of Agile practices, often described as "Agile-lite", generally results in suboptimal project outcomes. In contrast, a comprehensive application of practices such as stand-ups, sprint planning, and retrospectives provides structured opportunities for collaboration, enhances communication among team members, and contributes to improved project performance. These observations are consistent with the findings of Hummel et al., and the growing adoption of Agile approaches indicates that teams are carefully selecting practices suited to their specific project contexts, resulting in a more focused, efficient, and effective software development process.

References

[1] D. Dönmez, G. Grote, and S. Brusoni, "Routine interdependencies as a source of stability and flexibility: A study of agile software development teams," *Inf. Organ.*, vol. 26, pp. 63–83 (2016).

[2] N. Ozkan, M. S. Gök, and B. Ö. Köse, "Towards a better understanding of agile mindset by using principles of agile methods," in *Proc. 15th Conf. Comput. Sci. Inf. Syst. (FedCSIS)*, Sofia, Bulgaria, pp. 721–730 (Sep. 2020).

[3] S. Dorairaj, J. Noble, and P. Malik, "Effective communication in distributed agile software development teams," in *Int. Conf. Agile Softw. Dev.*, Berlin, Germany: Springer, pp. 102–116 (2011).

[4] R. Sandstø and C. Reme-Ness, "Agile practices and impacts on project success," *J. Eng. Proj. Prod. Manag.*, vol. 11, no. 3, pp. 255–262 (2021).

[5] C. Baham and R. Hirschheim, "Issues, challenges, and a proposed theoretical core of agile software development research," *Inf. Syst. J.*, vol. 32, pp. 103–129 (2021).

[6] M. Pikkarainen, J. Haikara, O. Salo, P. Abrahamsson, and J. Still, "The impact of agile practices on communication in software development," *Empir. Softw. Eng.*, vol. 13, no. 3, pp. 303–337 (2008).

[7] P. Abrahamsson, J. Warsta, M. Siponen, and J. Ronkainen, "New directions on agile methods: A comparative analysis," in *Proc. 25th Int. Conf. Softw. Eng.*, Portland, OR, USA, pp. 244–254 (May 2003).

[8] B. Boehm and R. Turner, *Balancing Agility and Discipline: A Guide for the Perplexed*. Boston, MA, USA: Addison-Wesley (2003).

[9] G. Arcos-Medina and D. Mauricio, "Identifying factors influencing on agile practices for software development," *J. Inf. Organ. Sci.*, vol. 44, pp. 1–31 (2020).

[10] M. Hummel, C. Rosenkranz, and R. Holten, "The role of communication in agile systems development," *Bus. Inf. Syst. Eng.*, vol. 5, pp. 343–355 (2013).

[11] T. Chow and D.-B. Cao, "A survey study of critical success factors in agile software projects," *J. Syst. Softw.*, vol. 81, no. 6, pp. 961–971 (2008).

[12] S. C. Misra, V. Kumar, and U. Kumar, "Identifying some important success factors in adopting agile software development practices," *J. Syst. Softw.*, vol. 82, no. 11, pp. 1869–1890 (2009).

[13] C. Tam, E. J. D. C. Moura, T. Oliveira, and J. Varajão, "The factors influencing the success of on-going agile software development projects," *Int. J. Proj. Manag.*, vol. 38, no. 3, pp. 165–176 (2020).

[14] R. Taylor, "Interpretation of the correlation coefficient: A basic review," *J. Diagn. Med. Sonogr.*, vol. 6, no. 1, pp. 35–39 (1990).

[15] H. Jeong, S. P. Mason, A. Barabasi, and Z. N. Oltvai, "Lethality and centrality in protein networks," *Nature*, vol. 411, pp. 41–42 (2001).

[16] D. Liu and Z. Zhai, "An empirical study of agile planning critical success factors," Master's thesis, Blekinge Inst. Technol., Karlskrona, Sweden (2017).

[17] A. Moniruzzaman and D. S. A. Hossain, "Comparative study on agile software development methodologies," *arXiv preprint*, arXiv:1307.3356 (2013).

[18] L. Cao and B. Ramesh, "Agile requirements engineering practices: An empirical study," *IEEE Softw.*, vol. 25, no. 1, pp. 60–67 (2008).

[19] D. Ghimire, S. Charters, and S. Gibbs, "Scaling agile software development approach in government organization in New Zealand," in *Proc. 3rd Int. Conf. Softw. Eng. Inf. Manag.*, Sydney, Australia (Jan. 2020).

Received November, 2024