Applying Scrumban in Hybrid Software and Hardware Teams: An Experience Report

Maria Alcimar Costa Meireles
Federal University of Amazonas
Instituto de Desenvolvimento Tecnológico - INDT
Manaus, AM, Brazil
maria.meireles@indt.org.br

Karen Soares
Instituto de Desenvolvimento Tecnológico-INDT
Manaus, AM, Brazil
karen.soares@indt.org.br

Ádria Afonso
Instituto de Desenvolvimento Tecnológico-INDT
Manaus, AM, Brazil
adria.afonso@indt.org.br

Lívia Lobão de Araújo
Instituto de Desenvolvimento Tecnológico-INDT
Manaus, AM, Brazil
livia.araujo@indt.org.br

Luana Lobão
Instituto de Desenvolvimento Tecnológico-INDT
Manaus, AM, Brazil
luana.lobao@indt.org.br

Gabriel Pinheiro
Instituto de Desenvolvimento Tecnológico-INDT
Manaus, AM, Brazil
gabriel.campos@indt.org.br

ABSTRACT

Context: Agile software development methodologies, such as Scrum, Kanban, and Scrumban, have gained prominence in Software Engineering by promoting greater flexibility and workflow visibility. Scrumban, in particular, combines elements of Scrum and Kanban and represents a promising alternative in contexts that demand continuous adaptation, such as hybrid projects integrating software and hardware development. **Goal:** This study investigates the perception of a multidisciplinary team regarding the application of Scrumban in hybrid projects involving simultaneous software and hardware development. **Method:** We conducted a survey with 15 team members from a Technological Development Institute to gather perceptions about using Scrumban in a hybrid context. We qualitatively analyzed the responses to identify reported benefits and challenges. **Results:** The findings suggest that Scrumban significantly enhances the visibility of task progress and fosters alignment among technical teams through visual boards and communication ceremonies. However, participants reported challenges in applying the methodology to hardware development, such as difficulties in task estimation, supplier dependencies, and the rigidity of physical development cycles. They emphasized the need to adapt agile practices to address the specificities of the hybrid context. **Conclusion:** The experience demonstrated that Scrumban is a valuable approach to promote integration and transparency in hybrid projects, although it requires adjustments to accommodate the particularities of hardware development. This report reinforces the importance of flexible and adaptive methodologies in multidisciplinary environments.

KEYWORDS

Scrumban, Hybrid teams, Agile methodologies

1 Introduction

In recent years, software development teams have widely adopted agile development as a standard approach, driven by the values and principles of the Agile Manifesto [5]. Teams have employed methodologies such as Scrum, Kanban, and more recently, Scrumban, a hybrid approach that combines elements of both to improve project management, enabling continuous delivery, and maximizing team efficiency through workflow visualization [5]. However, when teams adopt these practices improperly, they may face increased waste, longer development cycles, and reduced product quality [2].

Although initially designed for the software industry, agile development has gradually found application in hardware projects, primarily due to its advantages in dealing with volatile, uncertain, complex, and ambiguous scenarios [12]. However, directly transferring agile principles to the hardware domain poses specific challenges, as complications hinder the full implementation of agility in this context [12].

Projects that involve simultaneous software and hardware development face specific challenges when exposed to traditional agile approaches [3]. Previous studies have deepened the understanding of these obstacles [3, 6], highlighting four main categories of challenges: (i) product-related physical limitations, such as stakeholders' reluctance to provide feedback on non-functional prototypes, the shortage of professionals with interdisciplinary knowledge, and the difficulty of coordinating deliveries of different natures; (ii) mindset, which involves individual and cultural resistance to agility, insufficient training, and conflicts between agile teams and traditional organizational structures; (iii) scaling, which refers to the difficulty of extending agility across multiple projects, including issues with knowledge transfer, inflexible organizational structures, and leadership resistance; and (iv) team distribution, including communication barriers, cultural differences, and the lack of standardized tools [3]. These factors highlight the added complexity of incorporating agile practices into hybrid contexts beyond the traditional software development scope.

Given this scenario, Scrumban has emerged as a promising alternative for hybrid projects by offering greater flexibility, visibility, and traceability throughout the development process [4, 7]. This article presents a case study on applying Scrumban in projects by a multidisciplinary team at a Technology Development Institute. To collect data, we surveyed 15 members of an interdisciplinary team, aiming to understand their perceptions of the benefits and challenges of adopting Scrumban in initiatives that combine software and hardware development.

We analyzed the collected data qualitatively and found that using Scrumban helped increase task progress visibility and promoted alignment between technical teams through visual boards and communication ceremonies. However, participants also reported limitations, such as difficulties in estimating hardware tasks, external dependencies, and the rigidity of physical development cycles. They emphasized the need to adapt agile practices to address the particularities of hybrid contexts better. This experience reinforces the importance of flexible and adaptive methodologies in multidisciplinary environments and offers practical insights for professionals and organizations facing similar challenges.

2 Background and Related Work

2.1 Scrumban

Scrumban is characterized as a hybrid agile methodology that integrates practices from both Scrum and Kanban to promote greater flexibility and efficiency in project management [2]. According to Alqudah and Razali [2], this approach incorporates Scrum practices such as iterative planning, task prioritization, and continuous process improvement. At the same time, it adopts Kanban elements such as workflow visualization and Work in Progress (WIP) limits, which help enhance the flow and predictability of deliveries [2]. Additionally, Albarqi and Qureshi [1] emphasizes that Scrumban combines the benefits of both methodologies. It facilitates project management by adopting Scrum principles that promote team collaboration during fixed-length sprints and require cross-functional team members. It also incorporates Kanban mechanisms such as WIP limits, workflow monitoring, and cycle time measurement through visual boards.

Fuentes-Del-Burgo et al. [5] argue that teams can understand Scrumban as a transitional method between Scrum and more advanced development models. By integrating Kanban elements into the Scrum context, Scrumban provides new perspectives and enhances teams' management capabilities. This integration allows teams to adapt more effectively to project-specific needs by combining the structured cadence of Scrum—such as sprints, daily meetings, planning, review, and retrospectives—with Kanban principles, including visual boards, WIP limits, and a focus on continuous improvement [10]. By merging the strengths of both approaches, Scrumban aims to improve workflow management and support the constant delivery of value throughout the project lifecycle [10].

2.2 Related Work

Alqudah and Razali [2] investigated how teams form the Scrumban method by selecting and combining practices from Scrum and Kanban. The authors identified the factors that influence this combination and provided guidelines to support agile teams in adopting Scrumban appropriately. They conducted the study in two phases. In the first phase, they reviewed the literature to identify practices, principles, and distinctions between Scrum and Kanban. In the second phase, they conducted semi-structured interviews with seven agile methodology experts from different countries. They analyzed the collected data qualitatively using content analysis with NVivo software. The results revealed that the formation of Scrumban strongly depends on the context and the team's prior experience with Scrum and Kanban. The study identified several critical factors for selecting and combining practices, including method prescription, roles and responsibilities, adoption timing, team size, work batch size, requirements prioritization, feature size, lead time, cost, quality, and absence of technical practices. The findings indicated that Scrumban proved more effective than either Scrum or Kanban alone in reducing waste, improving quality, and saving time and costs by enabling flexible customization aligned with project characteristics.

Patil and Neve [7] investigated the practical application of Scrumban to enhance productivity in software development. Their study aimed to demonstrate how combining practices from Scrum and Kanban could mitigate limitations observed in traditional methodologies while promoting greater agility, quality, and efficiency in development processes. To this end, the authors conducted an experimental approach in which development teams adopted Scrumban by removing the fixed sprint cycles of Scrum and incorporating Kanban practices such as WIP limits and just-in-time planning. The analysis relied on evidence collected during the implementation, including communication records, productivity metrics, and observations regarding process improvements. The main benefits identified included increased flexibility to handle changes, improved visualization of the workflow, reduction of impediments, enhanced efficiency, and improvements in the quality of the final product. The authors also noted that Scrumban supports kaizen (continuous improvement) and waste reduction, contributing to technical and business gains. The study reinforces Scrumban's suitability for teams that require greater adaptability without compromising visibility and control over the development process.

Patilla et al. [8] explored how teams can adopt the Scrumban method to improve productivity in the software development process. The study followed a practical approach, implemented across multiple development teams in IT companies, and observed the impacts of Scrumban adoption in real-world environments. The methodology involved removing rigid aspects of Scrum, such as fixed sprints, and integrating Kanban practices, including WIP limitations, continuous prioritization, and workflow visualization. Field evidence was collected through direct observations of team performance and feedback from involved professionals, including software engineers and module leaders. The results indicated that forming a Scrumban process depends on teams' experience with agile methods and the project's specific needs. The study identified several critical factors for combining practices: planning flexibility, adjustment of work, batch sizes, delivery pace, team collaboration, and focus on continuous improvement. Scrumban led to improvements in areas such as lead time, process visibility, waste reduction, and delivery quality. Additionally, the methodology proved effective in adapting to change and delivering customer value, overcoming limitations observed in the isolated application of Scrum or Kanban.

Based on these related studies, it is evident that the existing body of research on Scrumban focuses predominantly on software development contexts, investigating its formation, benefits, and critical adoption factors in traditional agile environments [2, 7, 8]. The distinctive contribution of this paper lies in applying Scrumban to a hybrid context that simultaneously involves software and hardware development. This specific setting introduces additional challenges, such as synchronizing development cycles with differing natures, managing physical and logistical dependencies, and

adapting agile practices to accommodate less frequent and more rigid deliveries. Thus, our study contributes novel experimental evidence regarding adapting Scrumban to highly interdependent environments across technical domains, broadening the method's applicability and offering practical insights for teams engaged in multidisciplinary projects.

3 Process Conduction

The research strategy adopted in this work was a field study, combining exploratory case studies and archival studies. According to Stol and Fitzgerald [11], field studies are conducted in natural settings—environments that exist prior to the research initiative—which makes them particularly suitable for investigating specific instances of phenomena, events, people, teams, and systems. This approach enables researchers to gain a deeper understanding of what is occurring and how processes unfold, often producing descriptive and exploratory insights.

In this context, we report the experience of adopting and adapting the Scrumban methodology in hybrid software and hardware projects at the Institute for Technological Development, a non-profit organization dedicated to technological innovation and the execution of Research, Development, and Innovation (R&D&I) projects. The experience was carried out between 2022 and 2024, with the primary goal of enhancing integration among technical teams and improving progress visibility in complex projects characterized by high interdependence across areas.

3.1 Organizational Context and Motivation

The Institute carries out projects for the industry, covering embedded systems, software applications, and integration with sensors, actuators, and physical devices. Multidisciplinary teams work on these projects: software and hardware developers, automation engineers, mechanical designers, testing specialists, and product owners (PO).

Before introducing Scrumban, the teams followed different methodological approaches: software team already applied agile methods, mainly Scrum, while hardware team operated under more traditional planning cycles, with sequential schedules, low granularity, and centralized documentation. This disparity hindered alignment across areas, reduced project visibility, and complicated joint delivery planning.

As the number of projects with hardware-software interdependence increased, the teams identified the need for a management approach that could support flexibility, predictability, and continuous collaboration. They adopted Scrumban as a viable alternative because it combines the structured rituals of Scrum with the continuous visualization and pull-based flow of Kanban.

3.2 Scrumban Implementation Steps

Below, we describe the main steps followed during the Scrumban implementation process to support its replication by other organizations operating in similar contexts:

(1) **Selection of the Pilot Project**: The team implemented the initial phase of Scrumban in a project that already had a defined Work Plan (WP) with a clear scope and schedule. This foundation allowed them to begin the experience with a certain degree of predictability, essential for the first adoption cycle.
(2) **Formation of a Multidisciplinary Team**: The team included softwares developers, familiar with Scrum and hardware engineers without experience in agile methods. This configuration enabled the team to quickly identify friction points between the two domains and propose practical adjustments.
(3) **Initial Assessment Using Planning Tools**: The team applied the CSD Matrix (Certainties, Suppositions, and Doubts) and the History Map to identify priority features, risks, and knowledge gaps. These tools helped align expectations between the team and management and supported the creation of an initial backlog.
(4) **Definition of Rituals and Initial Cadence:** The team established a biweekly sprint cadence and adopted standard Scrum rituals: planning, daily meetings, reviews, and retrospectives. These events structured progress tracking and aligned incrementais deliveries, particularly in software.
(5) **Customization of the Jira Board**: The team customized the Jira board with columns common to all areas (e.g., Development Stage, Test Stage, Review Stage - See Figure 1). However, they defined distinct internal flows for each type of task card to reflect the unique characteristics of software and hardware. For instance, hardware tasks did not necessarily follow all stages, while software tasks moved through the complete workflow.
(6) **Establishment of Weekly Integration Checkpoints**: In addition to the standard ceremonies, the team created a weekly meeting between representatives from the software and hardware areas. These checkpoints focused on discussing task dependencies, bottlenecks, and planning adjustments. This practice facilitated early conflict detection and dynamic coordination of activities.
(7) **Continuous Process Adaptation**: Over the months, the team iteratively adjusted the process. For example, they realized that shorts sprints did not suit the hardware team's pace, so they introduced more flexible cycles for that domain. They also improved the use of labels and filters in Jira to flag blocked tasks, dependencies, or those requiring external validation (e.g., supplier deliveries).
(8) **Promotion of a Collaborative Culture**: Alongside tools and ceremonies, the team consistently promoted a culture of collaboration. Engineers were encouraged to participate in meetings actively, understand the context of other areas, and contribute to joint solution-building. This cultural shift was essential to breaking the perception that "hardware cannot run agile."

3.3 Conditions for Replicating the Experience

Other companies or R&D centers seeking to replicate this experience should preferably ensure the following conditions:

- **Multidisciplinary and Interdependent Context**: Projects that require simultaneous technical deliveries in software

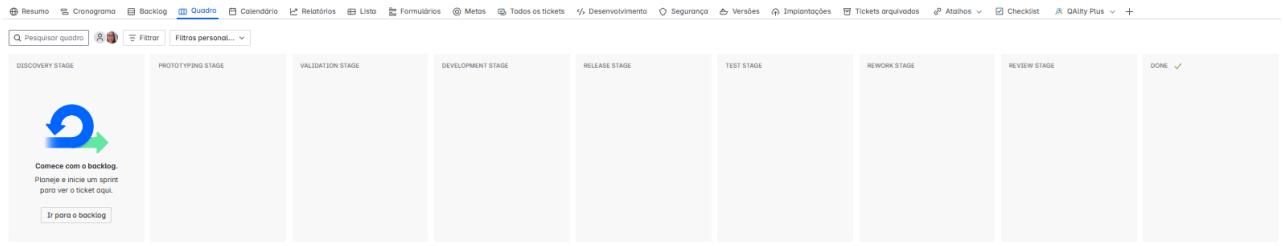

Figure 1: Customization of the Jira Board (in Portuguese)

and hardware or involve sensors, physical devices, and functional integration.

- **Management Tool with Customization Capabilities**: Using Jira or an equivalent tool that allows the definition of distinct workflows and the flexible application of filters, labels (for card categorization), and checkpoints.
- **Minimum Maturity in Agile Practices**: Having at least part of the team familiar with agile methods facilitates the transition and supports the initial adoption of the methodology.
- **Commitment from Technical Leadership**: Active involvement of the Product Owner, Technical Lead, and Management is essential to drive necessary adaptations over time, resolve impediments, and ensure strategic alignment.
- **Spaces for Continuous Learning**: Embracing ongoing adjustments to the process and staying open to testing new practices (such as different cadences between areas, integration meetings, and more) are crucial for Scrumban to succeed in hybrid contexts.

Following these steps and conditions, we believe teams with similar characteristics can benefit from using Scrumban as an adaptive methodology to foster greater integration, predictability, and collaboration in hybrid projects.

4 Qualitative Result

4.1 Participants

We conducted the study with 15 participants from a multidisciplinary team at an Instituto de Desenvolvimento Tecnológico (INDT). These professionals work in various areas, including software development (SW Dev), software development specialist (SW Specialist), mechanical engineering (Mechanical Dev), automation (Automation Dev), computer vision (Computer Vision Engineer), testing (Tester), and product owner (PO). This diversity of profiles enabled the collection of complementary perspectives on applying Scrumban in a hybrid development environment. The study was based on a survey composed of ten questions, which addressed perceptions, challenges, and suggestions related to the use of Scrumban. The complete list of questions is available in the supplementary material at link: [https://figshare.com/s/7e295f7d40e52055d6bc?file=56493779]. Table 1 presents the characterization of the participants involved in the study.

We asked participants to evaluate their overall experience with using Scrumban up to the time of the study. Figure 2 shows that 20% rated the experience as very positive, 46.7% as positive, 20% as neutral, and 13.3% as negative. These results indicate a generally favorable perception, though some reservations remain. In the following subsections, we present a qualitative analysis of participants' perceptions regarding using Scrumban in a hybrid context that involves hardware and software development.

Table 1: Characterization of Participants

Part	Role	Experience in Role	Years using Scrumban	N. Projects using Scrumban
P1	Sw Dev	3 years	3 years	5
P2	PO	7 years	6 years	10
P3	Tester	15 years	10 years	15
P4	Mechanical Dev	5 years	1 year	3
P5	SW Dev	3 years	2 years	6
P6	SW Dev	6 years	6 years	6
P7	Automation Dev	5 years	2 years	5
P8	SW Specialist	10 years	7 years	7
P9	PO	10 years	6 years	6
P10	SW Specialist	17 years	7 years	4
P11	Computer Vision Engineer	6 years	1 year	2
P12	SW Specialist	10 years	7 years	10
P13	SW Dev	6 years	2 years	3
P14	Automation Dev	5 years	1 year 6 months	4
P15	Automation Dev	10 months	10 months	3

4.2 Scrumban Adoption in Projects Integrating Software and Hardware

Applying Scrumban as a hybrid approach has proven a relevant alternative for teams working on projects that integrate software and hardware development. Overall, the study participants reported that the method is suitable, particularly because it balances practices from both Scrum and Kanban, allowing greater flexibility in planning and executing tasks. As observed by P1 and P6: "*very*

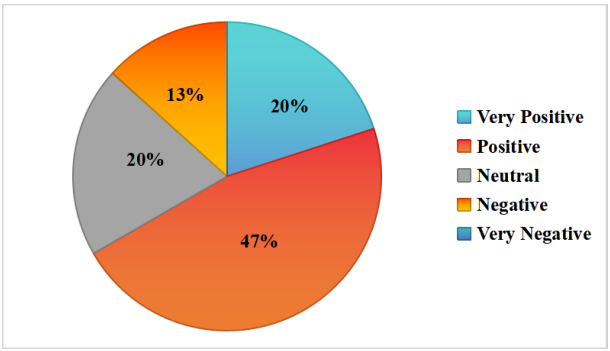

Figure 2: Overall experience with using Scrumban

suitable." P3 added, "*I believe slight improvements can be tested. But overall, very suitable*."

One of the main reported benefits concerns work visualization and organization. P2 says the methodology is "*very useful for tracking and visualizing activities*," while P12 sees it as essential for "*organizing priorities*." P5 emphasized the importance of communication enabled by Scrumban rituals: "*communication has been the main factor for this success. With its alignment rituals, the method ensures constant communication between the software and hardware team.*"

However, participants also pointed out limitations when applying the approach to the hardware context. P11 noted that "*hardware tasks face setbacks inherent to the physical nature of the components involved, such as purchase delays and the need to ensure compatibility between devices from different architectures, which may require extending a sprint's duration.*" P14 added, "*this does not cause major project impacts, but it shows that hardware's longer execution cycles do not always align with sprint timelines.*" P15 reinforced, "*the limitations are significant due to longer execution cycles, physical dependencies on components, and higher time costs for implementing changes.*"

Additionally, P4 questioned the effectiveness of Scrum rituals, stating that "*reviews carried out by the PO and the Technical Lead may not be well executed, as they do not always have the necessary technical expertise.*" Similarly, P7 argued that "*Scrum must be adjusted to the specificities of each project*," P8 observed that "*software and hardware have different needs.*"

Finally, P9 pointed out that the combination of rigidity and flexibility offered by Scrumban is especially useful for dealing with uncertainty in hardware projects: "*it allows the project to stay focused on the sprint goals and the planned scope, while still having the flexibility to handle changes that may arise along the way.*"

Thus, the qualitative data indicate that Scrumban is widely recognized as useful, primarily due to its adaptability. However, fully leveraging its effectiveness requires adjustments that account for the specific cycles and dependencies involved in hardware development and alignment of team roles and responsibilities.

4.3 Visibility of Task Progress Using Scrumban

Visibility of task progress is a feature frequently attributed to Scrumban and was highlighted by participants as one of its main perceived benefits. The combination of Kanban's visual board and Scrum rituals allows the team to continuously and collectively monitor ongoing activities. For example, P5 stated, "*visibility is one of the greatest benefits of Scrumban because its visual board displays the entire workflow to the team. This and daily meetings allow everyone to track progress in real time.*"

Other participants also emphasized the clarity and accessibility of information provided by the task board. P1 noted, "*it is very easy to see the status of activities, at which stage they are, and to estimate their completion*," while P2 added, "*we get a sense of everyone's development within the team.*" Alignment rituals strengthen collective understanding of project progress, as stated by P7: "*they help us understand the status of project activities through the team's shared knowledge during ceremonies and through continuous team alignment.*"

However, despite the general recognition of Scrumban's usefulness for improving visibility, some concerns emerged in the context of hardware development. P15 warned about the risk of misleading visibility: "*Scrumban can offer visibility, but without proper adaptations, it creates deceptive visibility that harms the team instead of helping.*" The participant illustrated this with an example task like "*sensor integration*," which may appear stalled on the board even though the developer is tackling multiple underlying technical challenges: "*the board shows the task 'stuck' for days, but in reality, the developer is solving multiple discovered technical issues.*" This disconnect between the visual task status and the actual effort involved can lead to unrealistic expectations, excessive focus on apparent productivity, and even team demotivation due to a perceived lack of progress. In this context, P7 reported, "*a challenge in hardware development is producing visible deliverables at the same frequency as the software team*," which may hinder stakeholder tracking. P6 further noted, "*visualizing the Kanban board becomes complicated when mixing agile development tasks with supplier dependencies or material delays*," which requires adjustments such as more refined control of the WIP.

Still, participants agreed that Scrumban can provide a useful and comprehensive view of workflow when properly implemented and adapted to the context. P11 pointed out that the ceremonies "*create room for an interdisciplinary perspective.*" At the same time, P14 emphasized that "*the meetings are essential for alignment, interaction, and understanding of the ongoing project, and they significantly support the team's progress.*"

Therefore, while participants widely acknowledged the visibility promoted by Scrumban, its effectiveness depends on the team's ability to adapt the methodology to the particularities of their tasks—especially regarding the complexities of hardware development. Creating mechanisms that capture actual task progress, beyond what is visually represented on the board, stands out as a practical recommendation for maximizing the benefits of this hybrid approach.

4.4 Challenges in Maintaining Discipline in Integrated Software and Hardware Development

Maintaining process discipline in teams that develop software and hardware simultaneously emerged as a multifaceted challenge in

adopting Scrumban. Based on participants' accounts, these challenges can be grouped into internal factors related to team practices and behaviors and external factors beyond the team's direct control. This distinction helps clarify how process discipline is affected in hybrid development environments and supports replicating mitigation strategies in similar contexts.

Internal Factors

Participants highlighted several internal issues related to team dynamics, organization, and workload management. A recurrent challenge involved resistance to task fragmentation, as mentioned by P9: *"The biggest challenge is getting those responsible for hardware activities to break these tasks into smaller, more micro parts, making them easier to measure."* P10 reinforced this view, noting that *"Kanban loses effectiveness in some areas if there is no more detailed schedule, such as measuring progress and anticipating risks."*

Another internal factor was the overload caused by status updates. P11 emphasized that *"the time spent on status reporting could be allocated to actual tasks."* At the same time, P14 added, *"the time consumed updating status could be used for real activities, considering that the updates are small fragments that do not represent the whole picture."*

Furthermore, forgetting to update the board also emerged as an internal issue (P12), revealing gaps in process adherence and monitoring practices. Although some participants, such as P4, reported that process discipline remained unchanged with Scrumban, most recognized the need for specific adaptations to fit the hardware context. P2 summarized this perspective: *"We need to adjust monitoring, especially for the hardware team, since their deliverables are not like those in software."*

External Factors

External influences were also identified as significant barriers to maintaining process discipline. One of the most frequent challenges was the misalignment of development paces between software and hardware. As P15 reported, *"the main challenge we face is the natural misalignment in development rhythms: while software follows the plan as expected and allows for less time-consuming adaptations, hardware has longer cycles due to its physical nature."* This difference creates ongoing pressure to reorganize workflows, especially when *"hardware blocks all development," requiring "emergency adaptations to the structured process"* (P15).

Participants cited external delays, such as supplier component deliveries, disrupting planning and coordination. P5 explained, "*The difficulty lies in maintaining discipline when external hardware dependencies, such as supplier delays, disrupt the planned workflow.*" At the same time, P7 observed that in hardware activities, *"it is harder to maintain the same frequency of reports or visible deliveries as in software activities, which creates an impression of delay or slowness not always aligned with actual progress.*"

Therefore, the reported experience shows that process discipline undergoes significant impacts in hybrid development environments, particularly due to external factors, differences in pace, physical constraints, and management practices. Future strategies to address these challenges should consider making the process more flexible, training teams to break down tasks, balancing visibility with workload, and redefining metrics that more accurately reflect progress in hardware development.

4.5 Challenges in Applying Scrumban in Hybrid Development Environments

Applying Scrumban in hybrid contexts that integrate software and hardware development has posed several challenges that directly affect the process's flow and effectiveness. Participants reported that the main complexity lies in reconciling the two domains' different paces, characteristics, and constraints.

One of the most frequent challenges involves the divergence between development cycles. While software operates iteratively and continuously, hardware depends on less flexible physical stages such as component acquisition, assembly, testing, and validation. P6 highlighted this difference: "*synchronizing distinct cycles is difficult—software allows continuous and iterative deliveries, while hardware depends on physical steps with less flexible timelines*." P15 also emphasized this asymmetry: "*change management becomes complex because software changes are agile, whereas hardware changes can take days or weeks for implementation, testing, and validation*."

Task estimation and planning also proved problematic, especially for the hardware team. The unpredictability of external factors, such as supplier delays and logistical constraints, directly affects deadline fulfillment. P12 noted, "*most of the time, the initial estimate does not match reality*," while P5 added, "*the reliance on external suppliers and the long lead times for physical prototype delivery make any hardware forecast complex*."

Another significant challenge involves the integration between software and hardware. P5 emphasized, "*the integration phase when compatibility issues between finalized software and newly arrived hardware emerge is a major obstacle*," which puts pressure on the teams and often leads them to loosen the structured process. In this context, P11 suggested creating mocking and simulation strategies as a possible mitigation: "*there is a tough trade-off between handling the highest-priority hardware tasks or unblocking the software development flow that depends on an urgent hardware delivery*."

Additionally, team maturity emerged as a limiting factor. P9 stated, "*there is no point in having a well-structured process if the team does not update the board or follow up properly*." P10 also identified the integration between software and hardware teams as a critical issue: "*team integration and the pipeline can suffer disruptions when dependent items are not planned*."

Finally, the very nature of hardware tasks presents challenges to the granularity required by agile approaches. P13 mentioned, "*hardware tasks are not always easy to break down*," which makes it harder to track them on the board and measure progress.

These accounts show that adopting Scrumban in hybrid environments requires specific adaptations, especially regarding cycle synchronization, delivery visibility, collaborative planning, and integration strategies across domains. Overcoming these challenges involves, among other factors, raising team maturity, creating practices that respect the specificities of each domain, and developing approaches that foster greater flow and collaboration between software and hardware.

4.6 Improvements in Software and Hardware Team Integration Through the Use of Scrumban

Participants identified the integration between software and hardware teams as a central issue, recognizing Scrumban's role as a facilitator and pointing out opportunities to make this integration more effective. They emphasized mutual visibility and constant communication as the key factors to improve alignment between the two areas.

P1 noted that the current model already brings important benefits by enabling parallel activity visualization: "*the parallel visualization of activities allows project management to gain more control and understanding of process progress, enabling more accurate deadline forecasting and quicker identification of blockers.*" The participant also emphasized that this visualization benefits team members by helping them identify interdependencies and maintain continuous alignment.

Along these lines, P10 reinforced the idea of a unified Kanban board that displays the dependencies between hardware and software: "*with a well-configured board, the project's progress flow becomes visible to everyone, allowing for control over hardware and software dependencies.*"

P2 also suggested the need for a more integrated project view. It recommended holding dedicated integration moments: "*there could be an integration session where the activities of both teams come together, just to gain visibility of the project as a whole.*" Similarly, P9 argued that Scrumban's hybrid model is promising for balancing flexibility and control but still needs improvements in visibility: "*I believe the main point to improve is exactly visibility making it clearer to the team what is being done and what is still to be done.*"

Other participants stressed that the process alone is not enough fostering a culture of cross-functional engagement is necessary. P7 stated, "*both teams should be more involved with the project as a whole and not just with their activities*," suggesting that understanding the other team's work can be as important as technical planning. P13 reinforced this idea by saying, "*one team should give visibility to the other about what they are doing.*" At the same time, P12 emphasized the importance of "*constant communication*" as a key element for integration.

Finally, P5 emphasized that a visual management system can support not only operational teams but also upper management in setting more realistic expectations: "*with a visual map of end-to-end work, it would be easier to justify to management and other departments why deadlines are longer*," helping to reduce external pressure and improve alignment across sectors.

In summary, the participants' accounts suggest that while Scrumban already contributes to team integration, it can be further strengthened through practices that enhance cross-team visibility, promote mutual understanding of tasks, and encourage continuous, context-aware communication in hybrid development environments.

4.7 Suggestions for Improving the Scrumban Process

Although most participants recognized Scrumban as a valuable tool to support the joint development of software and hardware, they offered several suggestions to make the process more efficient and better suited to the specific characteristics of the team's activities.

One of the main aspects mentioned was the adaptation of the Kanban board. Participants recommended adjusting the visualization and organization of activities to reflect the reality of the hybrid workflow better. P3 proposed a more thoughtful customization: "*I would rethink the board's customization rules for moving cards, the states assigned to each column, and improving the use of labels to identify tasks with extra demand, blockers, or dependencies on others.*" P6 added by suggesting the need to "*adapt the Kanban board to reflect hardware dependencies*," while P8 advocated for "*separating hardware and software activities or breaking them down.*"

Another recurring point involved the frequency and format of the ceremonies. P15 suggested reducing the number of meetings, explaining that "*status changes in the hardware team happen much more slowly*," while P4 recommended holding only "*one or two dailies per week at most.*" In line with these views, P7 emphasized that "*specialists should be more present during the ceremonies so that the concepts of 'Done' and 'Review' make sense during task development, minimizing rework and improving the planning of subsequent activities.*"

Participants also mentioned limitations in adopting the sprint model, especially for the hardware team. P12 was emphatic: "*The sprint model is not feasible in the hardware context*," a view shared by P14, who stated that "*hardware development becomes more difficult when using sprints.*" This challenge stems from longer cycles and lower predictability in physical deliverables, which often conflict with Scrum's structured sprint cadence.

Some participants pointed out the need to pay greater attention to stages related to the acquisition and logistics of physical components. P11 suggested, "*I would include item ordering/acquisition in the project pipeline*," reinforcing the importance of anticipating risk analysis tied to these dependencies.

Despite the suggestions, some professionals expressed satisfaction with the current process. P1, P5, and P13 stated they had no changes to propose, and P12 added that the process has been efficient precisely because "*it does not involve too much bureaucracy.*"

These accounts show that while Scrumban proves to be a promising approach in multidisciplinary environments, its effectiveness depends on contextual adaptations, especially regarding task visualization, ceremony adjustments, sprint flexibility, and the integration of logistics stages in hardware development.

5 Lessons Learned

Based on the data analyzed throughout this experience, we identified a set of lessons learned that can guide other teams interested in applying Scrumban in hybrid development contexts:

- **Adaptation is essential in hybrid contexts**: Although Scrumban offers an interesting combination of Sprint organization (Scrum) and flexibility (Kanban), applying it to projects integrating software and hardware requires adjustments. The divergence between agile software development cycles and the longer, more rigid cycles in hardware demands planning, ceremonies, and metrics adaptations to ensure that the process remains functional and sustainable.

- **Visibility must be contextualized**: Using visual boards and alignment ceremonies significantly enhances task visibility. However, complex and long-duration tasks may appear stagnant on the board in hardware projects even when substantial technical progress is made. This can lead to misinterpretations by the team and management, making it necessary to adopt mechanisms that reflect the effort invested in each task.
- **Software-hardware integration relies on collaboration, not just tools**: While Scrumban can promote greater integration between teams, the success of that integration depends on mutual engagement. Understanding the other team's activities, maintaining continuous communication, and building a shared vocabulary are as important as defining dependencies on the Kanban board.
- **Task granularity remains a persistent challenge in hardware**: Breaking down activities into smaller, traceable units is essential for maintaining the flow of the agile process. However, this remains difficult for hardware teams, whose tasks often involve sequential steps, physical dependencies, and logistical uncertainties. Investing in training to improve task detailing can enhance control and predictability.
- **External factors can compromise process discipline**: Process discipline faces barriers in hardware development, mainly due to supplier dependencies, logistical delays, and high costs associated with changes. These factors affect delivery regularity and require the process to remain flexible without losing focus on its objectives.
- **Not all Scrum practices apply equally to both domains** Sprints and structured ceremonies proved more suitable for software development than hardware. Shorts sprints and frequent meetings can be counterproductive for hardware, indicating the need for a dedicated cadence for each domain or the flexibilization of certain ceremonies.
- **Mapping and anticipating logistical risks is crucial**: As some participants proposed, including steps related to component acquisition and logistics directly in the workflow is essential for managing risks that directly impact hardware project progress. Anticipating these tasks in the pipeline can help reduce future bottlenecks.

6 Limitation

In this subsection, we discuss the limitations of qualitative studies based on Ralph et al. [9].

Our study presents some limitations inherent to qualitative research. First, we based our findings on a small number of participants, which limits statistical generalization. However, qualitative research does not aim for generalization, but rather for generating in-depth and contextualized insights. We ensured credibility by providing descriptions of the study's context, participant profiles, and procedures, allowing readers to assess the relevance of our findings to other settings.

Second, although the multidisciplinary composition of participants provided a rich perspective aligned with the reality of hybrid projects, the limited number of professionals involved and the predominance of internal members from Institute may restrict the generalizability of the findings. The absence of external teams or participants with varying maturity levels in agile methods may have influenced perceptions of Scrumban adoption. Future studies involving organizations with different cultures, structures, and levels of familiarity with hybrid approaches could broaden the understanding of the benefits and challenges of applying Scrumban in more diverse contexts.

Additionally, since this report is based on a single case study conducted at a specific technology development institute, the results reflect a particular experience within an organizational context that includes institutional support for experimenting with agile methods and a high level of technical specialization among teams. This uniqueness may limit the transferability of the findings to companies with more rigid structures, less team autonomy, or different levels of maturity in software and hardware engineering. Evaluations across multiple organizational contexts could strengthen the robustness and applicability of the reported lessons.

7 Final Considerations

This experience explored the application of the Scrumban method in a hybrid software and hardware development context, highlighting the benefits and the challenges inherent to this practice. The experience described in the study indicates that, although Scrumban is a versatile approach capable of promoting greater visibility, integration, and adaptability to change, its effectiveness depends on a set of specific adaptations to the characteristics of the hardware development environment, which typically involves longer cycles, stricter time constraints, and tasks that are less amenable to fragmentation.

One of the lessons learned concerns the need to customize the Kanban board, adjust movement rules, create columns that reflect hardware-specific dependencies, and differentiate software and hardware activities when necessary. The adaptation of ceremonies, such as reducing meeting frequency and ensuring the participation of specialists during review sessions, also proved important to avoid overload and ensure the effectiveness of collaborative practices. In addition, fragmenting hardware tasks was identified as a key strategy to improve traceability, monitoring, and progress predictability. However, it still faces cultural and operational barriers among those responsible for physical activities.

Another critical issue involves synchronizing distinct development cycles between software, which operates iteratively and continuously, and hardware, which depends on physical stages that require longer lead times. This asymmetry directly affects planning, change management, and the definition of realistic delivery goals, reinforcing the importance of collaborative strategies, expectation alignment, and more flexible management approaches. In this context, the high maturity of teams, both in technical expertise and in collaborative culture, also emerged as a determinant factor for the successful adoption of Scrumban.

The results revealed that applying Scrumban in contexts involving hardware activities requires specific adaptations, since some of its principles proved less effective under the physical constraints and external dependencies typical of this domain. The practices of short cadences, continuous Kanban updates, and progress measurement through incremental deliveries showed limitations, as

hardware development cycles are longer, subject to supplier delays, and less conducive to rapid iterations. These factors reduced process visibility and predictability, impacting discipline maintenance. Thus, the evidence suggests that Scrumban, while helpful in promoting integration and workflow alignment in hybrid teams, must be adjusted to account for the different rhythms and constraints of hardware development, incorporating complementary practices for planning, monitoring, and defining intermediate milestones.

Finally, despite the obstacles, the approach proved to be a promising tool, fostering a holistic view of workflow, enhancing interdisciplinary alignment, and providing continuous improvement mechanisms. The reported experiences suggest that the successful implementation of Scrumban in hybrid environments is directly related to an organization's ability to promote a culture of constant communication, transparency, and flexibility while investing in team training to address the specificities of each domain.

For future research, we are investigating more effective practices for tracking and measuring progress in hardware development, exploring synchronization strategies across heterogeneous workflows, and examining the influence of team culture and maturity on the success of agile practices in multidisciplinary projects. These insights may help define more concrete guidelines to enhance the benefits of Scrumban in complex environments involving the concurrent development of physical and digital products.

ARTIFACT AVAILABILITY

The dataset and material used in this research are currently maintained as an open-source project accessible at:

https://figshare.com/s/7e295f7d40e52055d6bc?file=56493779.

To avoid leakage of sensitive data and ensure privacy, we choose to anonymize all personal information provided in this paper.

ACKNOWLEDGMENTS

We thank all participants of the empirical study and INDT for their support. This work results from the Research, Development, and Innovation (RD&I), using resources from the Informatics Law for the Western Amazon (Federal Law 11.196/2005).

REFERENCES

[1] Aysha Abdullah Albarqi and Rizwan Qureshi. 2018. The proposed L-Scrumban methodology to improve the efficiency of agile software development. *International Journal of Information Engineering and Electronic Business* 11, 3 (2018), 23.

[2] Mashal Alqudah and Rozilawati Razali. 2018. An empirical study of Scrumban formation based on the selection of scrum and Kanban practices. *Int. J. Adv. Sci. Eng. Inf. Technol* 8, 6 (2018), 2315–2322.

[3] Alexander Atzberger and Kristin Paetzold. 2019. Current challenges of agile hardware development: What are still the pain points nowadays?. In *Proceedings of the design society: international conference on engineering design*, Vol. 1. Cambridge University Press, 2209–2218.

[4] Krunal Bhavsar, Vrutik Shah, and Samir Gopalan. 2020. Scrumbanfall: an agile integration of scrum and kanban with waterfall in software engineering. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)* 9, 4 (2020), 2075–2084.

[5] Joaquin Fuentes-Del-Burgo, Sebastián Pérez, and Miguel Ángel. 2022. 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*.

[6] Nis Ovesen and Chris Dowlen. 2012. The challenges of becoming agile–experiences from new product development in industry and design education. In *International Conference on Engineering and Product Design Education*. The Design Society.

[7] Saniav Pandit Patil and Jitesh R Neve. 2018. Productivity Improvement of Software Development Process Through Scrumban: A Practitioner's Approach. In *2018 International Conference On Advances in Communication and Computing Technology (ICACCT)*. IEEE, 314–318.

[8] Hubner Janampa Patilla, Judith Maribel Vilca Alviar, and Yudith Meneses Conislla. 2023. Scrumban/XP: Propuesta para mejorar la eficiencia de la gestión de proyectos ágiles en el desarrollo de software. *Revista Ibérica de Sistemas e Tecnologias de Informação* E61 (2023), 14–32.

[9] Paul Ralph, Rashina Hoda, and Christoph Treude. 2020. ACM SIGSOFT empirical standards. (2020).

[10] Victor Zamora Semerano and Luciano Francisco De Oliveira. 2024. O Papel Estratégico do Analista de Qualidade (QA) em Equipes SCRUM, Kanban e Scrumban no Desenvolvimento de Software Ágil. *Advances in Global Innovation & Technology* 2, 2 (2024), 32–45.

[11] Klaas-Jan Stol and Brian Fitzgerald. 2020. Guidelines for conducting software engineering research. In *Contemporary Empirical Methods in Software Engineering*. Springer, 27–62.

[12] Paweł Weichbroth. 2022. A case study on implementing agile techniques and practices: Rationale, benefits, barriers and business implications for hardware development. *Applied Sciences* 12, 17 (2022), 8457.