HAUTE ÉCOLE ALBERT JACQUARD

HEH.be Haute École en Hainaut

Haute École Condorcet

UMONS Université de Mons

UCLouvain

Département des industries créatives et numériques

Video Game Development in Higher Education: Project Management Experiences

Exploring Students Challenges and Adaptations

Mémoire présenté par

DUDON Adrien

En vue de l'obtention du diplôme de Master en Jeu vidéo

sous la supervision de

LADURON Christophe, superviseur

VANDEPOELE Godefroy, coach

Année académique 2024-2025

Table of Contents

Acknowledgements

The completion of this thesis would not have been possible without the invaluable assistance of my supervisor Christophe Laduron, my coach Godefroy Vandepoele, and Thomas Wostyn.

They provided me with valuable insights and resources throughout the year on project management and production, which also helped me and my team in the completion of my own video game project.

Their continuous support and assistance was greatly appreciated.

Introduction and problem statement

Higher education institutions commonly use practical projects as pedagogical tools. These projects not only help students acquire discipline-specific skills but also promote the development of project management skills, which are essential across nearly all professional domains (Holzbaur, 2010). In the field of software development, project management practices often align with Agile methodologies, with frameworks such as Scrum being widely implemented, though many alternatives exist (State of Agile, 2023). However, the implementation of these frameworks is not always successful. A lack of training is frequently cited as one of the primary reasons for failure, among other contributing factors (Majeed, 2012). The effective teaching of project management tools and methodologies can help mitigate these challenges, positioning higher education institutions as key actors in this process.

Software development commonly relies on Agile Project Management practices, which emerged in response to the limitations and challenges associated with traditional models such as Waterfall. Over time, these Agile methodologies have been refined and are now widely adopted by the majority of software development teams. While video game development is rooted in software engineering, it also belongs to the creative industries. The unique challenges of game production, particularly its high degree of unpredictability, make Agile methodologies particularly well-suited to game development, highlighting the importance of acquiring knowledge and training in Agile practices for professionals entering the video game industry.

Nevertheless, research indicates that applying project management methodologies in an academic setting introduces additional layers of complexity. Students face unique constraints and challenges not typically encountered in professional environments (Masood et al., 2018). These difficulties are further compounded in video game development courses, where student teams must coordinate across a wide range of disciplines, including animation, art, audio, programming, and UI/UX design (Chandler, 2020). Specifically, the studies from Masood only focus on the implementation of the Scrum framework by students.

While there is extensive literature on software project management, offering valuable insights for teams and organisations (Ahmed et al., 2014; Cho, 2010; Juricek, 2014; Majeed, 2012), video game development, though a form of software development, presents unique challenges and has also been the subject of significant research (Keith, 2020; Koutonen & Leppänen, 2013; McKenzie et al., 2021; Pałka, 2023; Ruonala, 2016).

However, to the best of my knowledge, there is a lack of research specifically addressing project management within student-led video game development projects in academic contexts. Prior studies emphasise the additional complexity inherent to video game development, which may intensify the challenges faced by student teams.

This study aims to explore the specific project management challenges encountered by multidisciplinary student teams, comprising game designers, artists, programmers, technical artists, and sound designers, while developing a video game from conception to completion. It further investigates the solutions these students implemented to overcome such challenges and adapt to the constraints of the academic environment.

To guide this investigation, the following research question was formulated:

“What specific challenges arise in managing a video game project within an academic setting for student teams, and what solutions are implemented to address these challenges and adapt to the academic environment?”

To address this research question, I conducted interviews with three student-led project teams, each of which had 1.5 years to develop a video game as part of a Master's program in video game development.

This thesis is structured into four main sections. First, the literature review explores current knowledge on project management, video game development, and the practices and challenges faced by student project teams. Next, the methodology section details the study's design, including the interview process and the questions asked. The results section then presents a summary of the interviews, focusing on the key and recurring themes identified across the teams. Finally, the discussion section examines the commonalities found in the interviews, offers my interpretations, and relates the findings to existing literature to provide further insight.

Literature Review

Software Development Practices

Traditional Software Development Practices

Traditional software development practices were largely dominated by the well-known Waterfall model. This model originates from the linear processes found in systems engineering and quality assurance disciplines of the 1950s and 1960s. While this sequential structure worked well for hardware development, it proved less suitable for software due to the intangible and often unpredictable nature of software products (Dybå et al., 2014).

Although Waterfall-style development had been practiced for some time before 1970, it was formally described for the first time by Royce (1970). Figure 1, originally published in Royce's paper, has since become one of the most cited representations of the Waterfall model.

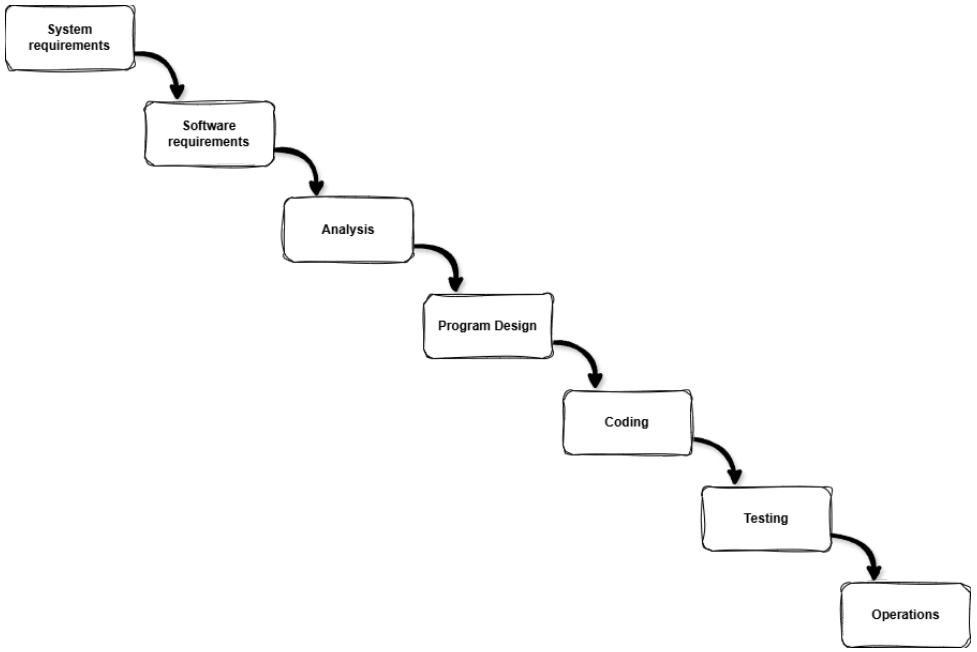

Figure 1. Software Development practices using a Waterfall-like model (Royce, 1970)

The model assumes that developers possess a near-perfect understanding of project goals and the final product from the start. As a result, it does not readily accommodate changes in scope, schedule, or resources once development is underway.

Interestingly, Royce never used the term Waterfall in his original paper. His work has frequently been misinterpreted as advocating for a strictly linear development process that assumes flawless execution at each stage (Leffingwell, 2007). While Royce was the first to formally illustrate a Waterfall-like model, he also warned against its limitations. He argued in favour of iterative development and emphasised the need for continuous customer involvement. (Royce, 1970, p. 2).

Among software developers, the Waterfall approach is widely recognised as prone to failure, particularly due to its inability to adapt to rapidly changing requirements and its tendency to result in budget overruns and project delays (Boehm, 2002; Boehm & Turner, 2003; Brooks, 1995; Schach, 2004; Sommerville, 2016; Watson et al., 1997).

Nevertheless, the Waterfall model still offers some advantages. Its formalised structure allows for greater transparency and consistency in project phases, making it easier for clients to monitor progress and maintain control. Clear definitions of responsibilities can enhance team cohesion and

the structured process allows for more accurate cost estimation and forecasting of financial outcomes (Pimonova, 2020). These benefits help explain why Waterfall-style methods continue to be employed in large-scale projects, particularly in the government sector (Cho, 2010).

Agile Development Practices

Agile Philosophy. Agile is a mindset of lightweight methods promoting the use of incremental and iterative methodologies to develop software. Agile is an umbrella term for a set of frameworks and practices, based on the Agile Manifesto for Software Development and the 12 principles behind it (Beck et al., 2001).

> We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
>
> **Individuals and interactions** over processes and tools
> **Working software** over comprehensive documentation
> **Customer collaboration** over contract negotiation
> **Responding to change** over following a plan
>
> That is, while there is value in the items on the right, we value the items on the left more.

"Agile Manifesto"

The Agile Manifesto is underpinned by several key principles, which emphasise that Agile methodologies are primarily user-centric, focusing on the end-user who will interact with the software or tool. An Agile team is characterised by its self-organising nature, taking full ownership of the development process and managing the details of development themselves.

Lean Principles. Lean is a set of principles originating from the manufacturing and industrial sectors, specifically from the Toyota Production System. This system introduced a novel approach to manufacturing logistics, which was later adapted for software development. At the core of the Toyota Production System is one of Lean's most fundamental principles: the elimination of waste (Poppendieck & Poppendieck, 2010).

Lean encompasses seven principles, all aimed at delivering value to users. These principles, outlined in the official "Lean Software Development" book (Poppendieck & Poppendieck, 2003) and are summarised below:

1. **Eliminate Waste:** Remove non-value-adding elements.
2. **Amplify Learning:** Involve players to understand their needs.
3. **Decide Late:** Delay decisions for better accuracy.
4. **Deliver Fast:** Use short cycles to reduce errors and add value quickly.
5. **Empower Teams:** Let teams control and own development.
6. **Build Integrity In:** Ensure cohesive, well-integrated design.
7. **See the Whole:** Understand system-wide interactions.

Lean software development is regarded as an Agile software development philosophy, emphasising efficiency, user value, and iterative progress.

Agile Development Frameworks

Scrum

Overview. Scrum is an Agile framework that incorporates Lean thinking and the principle of empiricism. Developed in the 1990s by Ken Schwaber and Jeff Sutherland, Scrum was formally

documented in the Scrum Guide in 2010. This thesis references the 2020 version of the framework (Schwaber & Sutherland, 2020). Scrum is an iterative and incremental process. Scrum teams are cross-functional and typically consist of no more than 10 members, as larger teams may face communication and coordination issues.

Within a Scrum team, three primary roles are defined:

- **Scrum Master:** Facilitates the adoption of Scrum and supports the team in following its principles through guidance and training.
- **Product Owner:** Manages the Product Backlog, prioritises tasks, and represents the interests of end-users and stakeholders.
- **Developers:** Encompass all team members contributing directly to the product, such as programmers, artists, designers, and others involved in development.

While it is possible for the Scrum Master or Product Owner to also be part of the development team, some studies caution against this, suggesting it may lead to conflicts of interest (Keith, 2020).

Events. Scrum defines five key events, including the overarching Sprint, which serves as a timeboxed container for four additional meetings. Sprints typically last between two and four weeks and are the primary unit of progress in Scrum.

- **Sprint Planning:** Held at the beginning of each Sprint, this meeting involves the entire team and aims to define the Sprint Goal, identify which Product Backlog items will be addressed, and determine how the work will be achieved.
- **Daily Scrum:** A 15-minute, timeboxed meeting held each day for developers to share progress, upcoming tasks, and any impediments.
- **Sprint Review:** Occurs at the end of the Sprint to inspect the outcome and determine whether the Sprint Goal has been met. It is an opportunity for the team to present work to stakeholders and gather feedback.
- **Sprint Retrospective:** The final meeting of the Sprint, involving only the Scrum team. It focuses on reflecting on the process, identifying what went well, what could be improved, and planning actionable improvements for the next Sprint.

Artefacts. Scrum artefacts represent work or value and provide transparency across the development process. There are three primary artefacts:

- **Product Backlog:** An ordered list of everything known to be needed in the product. Items are prioritised, with the most important items placed at the top.
- **Sprint Backlog:** A subset of the Product Backlog selected for a Sprint, along with a plan for delivering the work and achieving the Sprint Goal.
- **Definition of Done (DoD):** A shared understanding of what it means for work to be complete. A task is not considered done unless it meets the criteria outlined in the DoD.

Process. The Scrum process begins with Sprint Planning, where the team collaboratively selects a Sprint Goal: a single, clear objective that guides the Sprint. The Sprint Goal is supported by the Sprint Backlog, which includes the highest-priority Product Backlog items.

During the Sprint, developers meet daily for the Daily Scrum to share progress and coordinate efforts. At the end of the Sprint, the team holds a Sprint Review to evaluate the outcome and present work to the Product Owner and stakeholders. The Sprint concludes with a Sprint Retrospective, held without the Product Owner, to reflect on the Sprint and identify improvements for the next iteration.

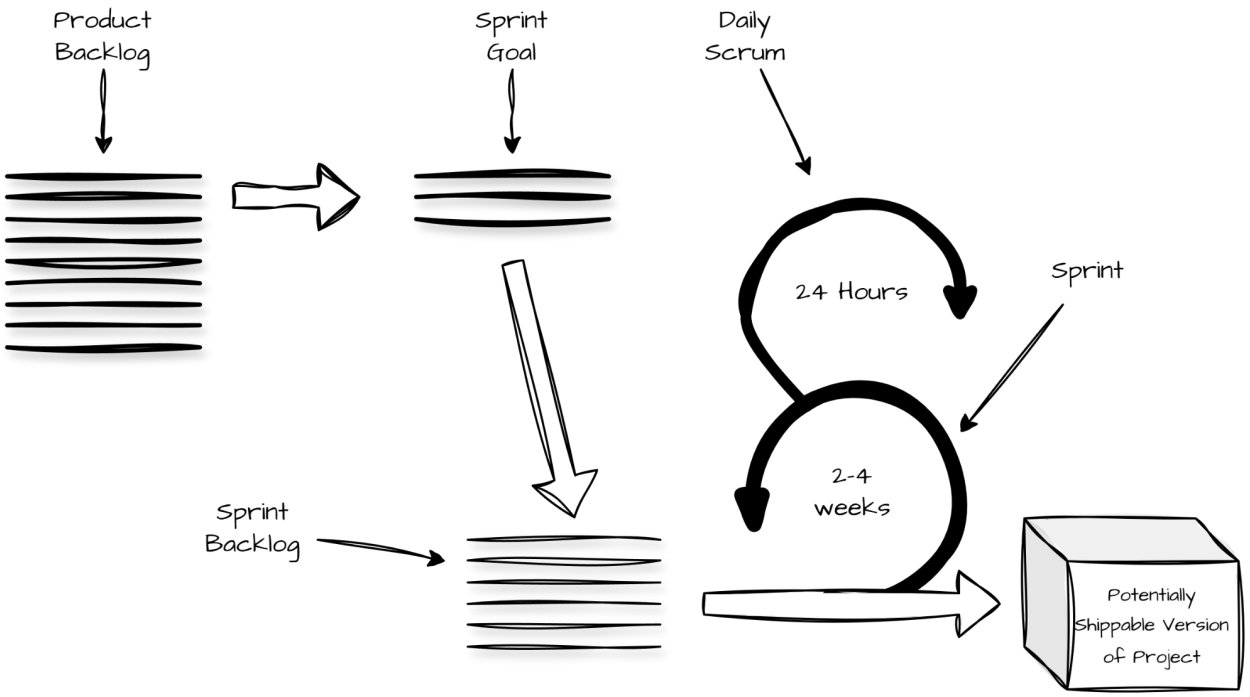

Figure 2. Scrum process – figure inspired by Keith (2020)

Kanban

Kanban is a visual, pull-based strategy aimed at optimising the flow of value through a process by defining and visualising workflows, actively managing work items, and continuously improving operational practices (Vacanti & Coleman, 2020). Rooted in flow theory and lean principles, Kanban balances effectiveness, efficiency, and predictability to enhance value delivery. It is adaptable across various industries and can complement existing project management methodologies.

The core practices of Kanban include:

1. **Defining and visualising workflows**: clearly outlining the steps involved in a process and specifying how tasks transition from one stage to another.
2. **Actively managing work items**: ensuring tasks are monitored and progressed in a timely and transparent manner.
3. **Improving workflows continuously**: using feedback and observation to make incremental adjustments that optimise the system.

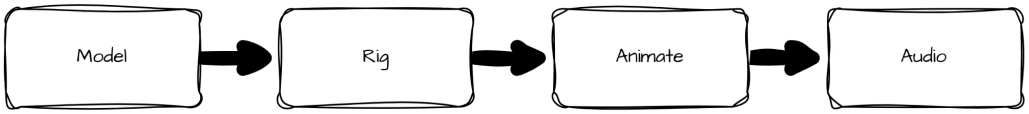

Figure 3. Flow of creating a character in a video game – figure inspired by Keith (2020)

A distinctive feature of Kanban is its emphasis on collecting and analysing metrics to monitor the "health" and performance of the workflow. These insights enable teams to identify bottlenecks and areas for improvement, fostering a culture of data-informed decision-making.

Unlike Scrum, Kanban does not prescribe a set of required and frequent meetings. Instead, it offers greater flexibility, allowing teams to determine the type, frequency, and purpose of meetings based on their specific needs. This flexibility makes Kanban particularly well-suited for support teams, maintenance workflows, or environments focused on bug tracking and resolution (Keith, 2020).

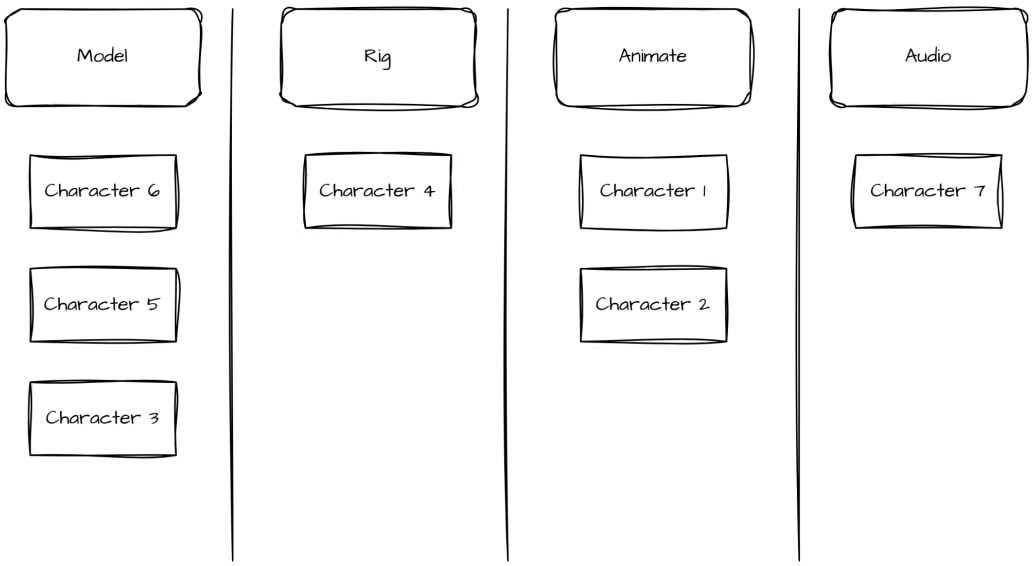

Figure 4. An example of a Kanban board – figure inspired by Keith (2020)

Scrumban

Scrumban is a hybrid methodology that combines elements of Kanban and Scrum to enhance project management by leveraging the strengths of both approaches (Alqudah & Razali, 2018). Kanban offers greater flexibility and freedom for team members, whereas Scrum emphasises structured iterations and defined roles. By blending these methods, Scrumban seeks to maintain Scrum's core principles, such as incremental development within time-boxed Sprints and the role of the Scrum Master, while incorporating Kanban's flexibility.

Unlike Scrum and Kanban, Scrumban lacks a formal set of practices or an official guide. Instead, teams select elements from both frameworks that best suit their needs, effectively creating a customised development process. An empirical study by Alqudah and Razali (2018) demonstrated that Scrumban can improve product quality, reduce time, and minimise waste more effectively than either Scrum or Kanban alone. However, their findings also highlighted the importance of supporting Agile teams in selecting the appropriate combination of practices to achieve these benefits.

It is important to note that the Scrum Guide emphasises that Scrum may be ineffective or counterproductive if its practices are not fully implemented (Schwaber & Sutherland, 2020). In this regard, Scrumban deviates from Scrum's strict guidelines by allowing selective adoption of practices from both Scrum and Kanban.

Feature Driven Development (FDD)

Feature-Driven Development (FDD) is a model-driven, short-iteration process that focuses on delivering working software through a sequence of well-defined, client-valued features. FDD was formalised by Coad et al. (1999) in a book that integrated Java and UML as foundational tools for modelling and implementation.

The process is structured around five core processes:

1. **Develop an Overall Model:** develop a high-level understanding of the future software.
2. **Build a Features List:** identify and categorise the features required by the software.
3. **Plan by Feature**: prioritise and assign features to Chief Programmers, which coordinates small development teams.

4. **Design by Feature**: sessions with small development teams are organised to define how a feature will be implemented
5. **Build by Feature**:The designed feature is implemented, tested, and integrated.

It also defines six core roles:

- **Project Manager (PM):** Oversees the administrative aspects of the project
- **Chief Architect (CA):** Owns the overall design and has authority over architectural decisions.
- **Development Manager:** Handles day-to-day development operations
- **Chief Programmers:** Senior developers who lead feature teams, participate in high-level analysis and design, and ensure delivery quality.
- **Class Owners:** Developers responsible for specific classes in the codebase.
- **Domain Experts:** Clients, users, sponsors, or analysts who provide the business knowledge necessary for accurate modelling and feature definition.

Additional roles may be added for large teams (e.g. Testers, Build Engineers, …).

Compared to methodologies like Scrum, FDD places a stronger emphasis on up-front modelling and structured roles, making it particularly well-suited for large or complex software systems that benefit from clear planning and domain understanding.

Methodology	Type	Structure	Best Use	Weakness
Waterfall	Sequential, plan-driven	Linear phases	Fixed-scope, large projects	Poor adaptability, costly changes
Scrum	Iterative, incremental	Timeboxed Sprints	Fast-paced, cross-functional teams	Can fail if incomplete, requires time and focus
Kanban	Flow-based, pull-driven	Visual workflow, WIP	Well-established workflow	Lacks role/event structure.
Scrumban	Hybrid (Scrum + Kanban)	Custom/partial Scrum	Balanced teams need flexibility	No official structure, can be easily misused
Feature-Driven Development (FDD)	Feature-based, iterative	5-step feature cycle	Large, domain-driven systems	Heavy upfront modelling, complex roles

Table 1: Comparative table of five project management methodology

Video Games Development

According to Esposito (2005), a video game can be academically defined as: *“A game which we play thanks to an audiovisual apparatus and which can be based on a story.”* In simpler terms, a video game is an interactive activity governed by rules, in which a player engages using electronic devices such as screens, controllers, or loudspeakers, and which may or may not include a narrative component. The Oxford English Dictionary further simplifies this notion, describing a video game as a form of moving images on a screen that players can influence through button presses.

The video game industry has seen continual growth over the past decades. In 2024, the global number of players is projected to reach 3.759 billion, an increase of 485 million compared to 2023, emphasising the sector's significance and the growing need for academic research in this domain (Semenchenko et al., 2025).

While large-scale AAA games are well-known, the majority of contemporary releases originate from independent (indie) game studios. As of 2024, indie studios developed 99% of the games released on Steam. In some cases, the production quality of these independent titles can rival or even surpass that of games from AA or AAA studios, blurring traditional industry boundaries.

This evolving landscape highlights the critical importance of effective project management methodologies. Given the iterative nature of game development, the Agile philosophy has become increasingly relevant. Aspiring game developers must receive proper training and hands-on experience with Agile methodologies to remain competitive in the industry.

Although Agile principles were originally conceived as a response to the rigid Waterfall model in software development, video game development presents unique challenges. While video games are indeed software, they are also creative products, involving a multidisciplinary effort that includes programmers, game designers, sound designers, narrative designers, artists, animators, and actors, among others. Agile methodologies, perceived as flexible by programmers, may be experienced as rigid or misaligned by artists or other creatives, complicating their adoption across disciplines (Engström et al., 2018; Hodgson & Briand, 2013).

Beyond technical and creative demands, studio culture, team dynamics, and project management practices also contribute to the challenges developers face (Keith, 2020; Pałka, 2023).

Keith identifies three primary difficulties in game development: feature creep, overoptimistic schedule and production issues. Although these difficulties may also appear in other software contexts, the inherently creative and multidisciplinary nature of video game development introduces unique challenges that further complicate the production process.

Agile Implementation Challenges

Agile methodologies such as Scrum, Kanban, and Feature-Driven Development (FDD) are widely adopted, but not without implementation challenges. These challenges can vary depending on the methodology and organisational context.

In the case of Scrum, Pałka (2023) categorises the main obstacles into three groups: people-related challenges, methodological understanding, and product delivery issues.

People-related challenges refer to difficulties arising from the team composition and logistics. For instance, geographical distribution can make it difficult to coordinate meetings, particularly the daily stand-ups that Scrum requires.

Methodology understanding challenges often involves a lack of clarity about the goals of a sprint or insufficient familiarity with the Scrum framework itself (Majeed, 2012). According to the Scrum Guide, teams are expected to be self-organising and self-managing. However, not all developers are comfortable with such autonomy and may prefer a more traditional, leader-driven approach (Pałka, 2023).

Product delivery challenges occur when planned increments are not delivered on time, which can erode the team's confidence in the methodology (Cho, 2010).

Kanban, while simpler and more flexible than Scrum, presents its own set of difficulties. Its minimal structure can be both a strength and a limitation. Studies suggest that Kanban often needs to be supplemented with additional practices to be effective, as seen in the hybrid approach of Scrumban (Alaidaros et al., 2021). Originating from industrial production and assembly line systems, Kanban is vulnerable to breakdowns in workflow continuity, if a single step fails, the entire process can be disrupted, requiring additional work to recover.

Agile Usage in an Academic Setting

Scrum, Kanban, and other Agile methodologies require proper training to be implemented effectively, yet research consistently shows that such training is often insufficient (Alaidaros et al., 2021; Cho, 2010; Pałka, 2023; State of Agile, 2023). This underscores the importance of incorporating Agile methodologies into higher education curricula, especially as the majority of software companies now use Agile practices.

Today students are increasingly exposed to Agile frameworks during their academic studies, particularly in computer science and related fields (Holvikivi, 2020; Masood et al., 2018; Tatnall, 2020). These educational experiences can be theoretical or involve practical applications through team-based projects.

Kropp and Meier (2013) conducted an experiment aimed at understanding why many graduates lack sufficient knowledge of Agile methodologies. Their motivation stemmed from an earlier survey in which companies reported that recent graduates were inadequately prepared in Agile practices. In response, they designed a course where students developed a 2D video game using Scrum. To simulate a realistic Scrum environment, each team elected a Scrum Master from among their members, while the lecturer acted as the Product Owner. The results indicated that the students responded positively to the course format and gained significant knowledge of Scrum through the experience.

However, the implementation of Agile methodologies within academic settings presents its own unique set of challenges, challenges that may, in some cases, exceed those faced in professional environments. A study by Masood et al. (2018) identified several of these difficulties based on survey data collected from students engaged in Agile projects. The most common issues reported included schedule constraints, poor team communication, limited customer interaction, lack of commitment, personal obligations outside school, and technical limitations. These findings highlight that while academic environments offer valuable opportunities to teach Agile practices, they also require thoughtful adaptation to address the specific needs and limitations of student teams.

Methodology

Research Objectives and Design

This study aims to explore the experiences of students developing a video game as their primary academic project. It focuses on the project management practices they chose, the reasons behind these choices, the adaptations required to fit the academic context, the challenges encountered during the process, and the role played by academic institutions.

A qualitative approach was selected due to the empirical nature of the study. The aim is to understand individuals' behaviours within a specific context and the factors that led to those behaviours (Oun & Bach, 2014). As Erickson (2024, p. 33) notes, "*qualitative research seeks to discover and to describe narratively what particular people do in their everyday lives and what their actions mean to them".* In this case, the participants' "*everyday lives*" refer to their work on the video game project, and their actions involve the decisions and practices implemented throughout development. Accordingly, a qualitative approach was deemed the most suitable method for gathering detailed empirical data.

Data Collection

The primary research instrument for this study was semi-structured interviews, conducted during the final milestone of each student team's project.

This case study focuses on three student teams enrolled in a Master's program in video game development at a higher education institution, the Haute Ecole Albert Jacquard (HEAJ) located in Belgium. Each team aimed to develop a small-to-medium-scale video game over a period of one and a half years.

Interviews Design

Each team included one producer. While the definition of the producer's role varied across teams, it was generally accepted that the producer acted as the project manager and was responsible for helping the team implement the chosen development framework.

The interview process consisted of two parts: a one-on-one interview with the producer, followed by a focus group with the remaining team members, conducted without the producer present. This structure was based on two key assumptions. First, the producer likely had the greatest influence on the selection and implementation of the project management framework, thereby shaping the team's collective approach. Second, the producer was assumed to be the most knowledgeable team member regarding project management practices and agile methodologies.

Conducting the interviews separately was intended to enhance the trustworthiness of the data and reduce potential biases. By excluding the producer from the focus group, team members could speak more freely and openly about their experiences and perspectives.

The Agile Manifesto emphasises that software development, and by extension, video game development, is inherently a collaborative effort, placing high value on the interactions among team members (Beck et al., 2001). The focus group interviews were therefore designed to explore how the team collectively perceived the development framework in practice, and how those perceptions aligned (or conflicted) with the producer's stated intentions. This allowed for a more nuanced understanding of the team's shared experiences.

As Dimitriadis and Kamberelis (2024) argue, *"real world problems cannot be solved by individuals alone, but require rich and complex funds of communal knowledge and practice."* Agile practices are not shaped solely by the producer or project manager but by the entire team.

Consequently, the use of focus groups as a data collection method was well suited to the collaborative nature of the development process examined in this research.

The initial interview questions were adapted from McKenzie et al. (2021), who investigated how and why video game studios adopt (or choose not to adopt) specific agile frameworks. In contrast to their study, which involved interviewing senior developers in leadership positions within professional studios, this research focuses on student teams developing a video game in an academic setting. As the participants are students rather than professionals, none are assumed to hold senior roles. However, the producer is presumed to have a greater understanding of project management practices and to have played a key role in selecting and implementing the development framework.

In this study, the term development framework is used to refer to the methodology or practices used by each team. Standard frameworks such as Scrum, Kanban, and FDD were included in the interview design. However, teams were not limited to these models. In practice, each team may have adapted or combined elements of multiple methodologies, resulting in a custom framework tailored to their specific needs.

The questions used for this interview are listed below:

Contextualisation:

1. What can you tell me about the composition and size of your team?
2. What development methodology did you use at the start of your project? Why did you choose that methodology over others?
3. Can you describe your initial development framework? Is this development framework a standard implementation of Scrum, Kanban, FDD, a hybrid, or something else?
4. What do you think was the general comprehension of the team regarding the chosen development framework?

Adapting the framework:

1. How did balancing coursework and other academic projects affect your ability to follow the framework?
2. How did time constraints or limited availability of team members impact your adherence to the framework?
3. What were the biggest challenges your team faced while adhering to the framework?
4. Did your team make any modifications or adjustments to the framework to suit your specific needs? If so, what were they?
    - Describe the development framework you are using now, compared to the one you initially choose to follow.
    - How did these modifications affect the balance between your project and your coursework?
5. Did the composition or size of your team have an impact on the adaptation made to the development framework? Why?

Team Dynamics and Communication:

1. How did your development framework influence communication and collaboration within the team?
2. Were there any moments where communication or teamwork broke down? Why? What role did the project management framework play in this?

Learning Outcomes:

1. If you were to start this project again, what changes would you make to your approach or the project management framework you used?
2. Were there any aspects of the project management framework that you wish you had explored or used more?

Final Reflection:

1. Did your team felt supported in implementing the framework? What resources (or lack thereof) impacted your ability to adhere to it? It can be support from the school or something else.
2. Overall, how effective do you think your project management framework was for your team's needs?
3. What advice would you give to future student teams?

Interview protocol

Interviews were conducted in person whenever possible; otherwise, they were held remotely via video conferencing. At the beginning of each session, participants were informed of the purpose and structure of the interview and given the opportunity to ask questions.

All interviews were audio recorded, with video recordings made during remote sessions. In addition, written notes were taken during each interview. Summarised notes and recordings were made available to participants for review to ensure the accuracy of their responses. Each interview lasted approximately 60 minutes.

To preserve confidentiality and promote honest responses, the content of each interview was not shared between participants. In particular, the responses of the producer were not disclosed to their respective team members, and vice versa, to avoid potential conflicts and response biases.

Interview analysis

The General Inductive Approach (GIA) was employed for the qualitative analysis of the interview data (Thomas, 2006). This method was specifically developed to provide "*a straightforward set of procedures to follow without having to learn the underlying philosophy and technical language associated with many qualitative analysis approaches*".

GIA was selected for its accessibility and practicality, as well as its strong foundation in grounded theory, discourse analysis, and phenomenology. It offers a clear and efficient analytical process, making it especially suitable for applied research. Additionally, this approach was used by McKenzie et al. (2021), the primary study that informed the design of this research, where it proved effective in analysing interview data. Given the similarity between the two studies, adopting the same method was a logical choice.

The GIA process involves three key steps:

1. **Condensing raw textual data** into a summarised format by identifying recurring concepts and categorising them through coding.
2. **Establishing clear links** between the research questions and the summarised findings derived from the data.
3. **Developing a model or framework** to describe the underlying structure of participants' experiences.

The interview data were coded using QualCoder, a free and open-source software tool, which facilitated the organisation and summarisation of the data presented in the Results section of this paper.

Results

All interviewed students are in the second year of a Master's program in Video Game Development. This program comprises four distinct tracks: Game Design, Programming, Technical Art, and Game Art.

The students began their project during the second term of their first year, resulting in a total duration of 1.5 academic years. All teams followed a shared curriculum consisting of general and specialisation courses based on their chosen tracks. Each student also received an introductory course in Project Management, with a focus on Agile philosophy and the Scrum methodology.

Each team was tasked with creating a complete video game, simulating the workflow of a professional video game studio. Projects covered aspects such as marketing, financing, studio structure, and team management. Each team was supervised by a faculty member with current experience in the video game industry.

During the first two academic terms, students had additional coursework not directly related to this project. As a result, teams could not devote their full time to development. This constraint, discussed later in detail, influenced team progress. Nevertheless, it is particularly noteworthy how teams adapted to this academic context, one that differs significantly from that of a professional studio.

The following sections present the interview results, summarising findings from the three student teams. Given the limited dataset and the distinct approaches taken, results are structured as follows:

- **Team Composition:** Details the initial and current make-up of each team, including the number of members, role distribution, and rationale for role assignments.
- **Team Dynamics:** Explores interpersonal dynamics, member departures, and role balancing.
- **Methodology:** Describes the development methodologies used by each team and how these evolved.
- **Communication:** Highlights challenges related to team communication.
- **Academic Constraints:** Examines constraints imposed by academic responsibilities and their impact on the teams.
- **School Support:** Reviews the types of institutional support provided and teams' perspectives on this support.

Team A

Team Composition

Team A comprised 10 members, making it relatively large for an academic project. Most members held hybrid roles, covering at least two disciplines. This increased the complexity of project management, requiring effective delegation and distribution of responsibilities.

A major challenge early in the project was an imbalance in team roles. Initially, the team adhered strictly to the Master's program four tracks, which led to a disproportionate number of Game Designers (five out of ten members). This created conflict during brainstorming sessions, where the differing creative visions of five game designers caused decision issues. As their producer explained:

> *Mais quand tu as cinq game designers qui discutent, ouais, c'est bon, tu n'as strictement rien du tout, parce que tout le monde a sa propre idée, tout le monde pense que cette mécanique de*

jeu est meilleure, et on doit essayer de se convaincre l'un et l'autre que ça c'est la meilleure idée.

(But when you have five game designers discussing, well, you get absolutely nothing, because everyone has their own idea, everyone thinks their game mechanic is better, and we have to convince each other that this is the best idea.)

In response, the team restructured their roles. Only one member retained a general Game Designer position, while others assumed more specific roles, such as Level Designer or Narrative Designer.

The team favoured a horizontal structure, distributing leadership across several team leads rather than concentrating authority on the producer. This approach enabled flexibility and ensured continuity when members were unavailable.

Ultimately, three key leadership roles emerged: Producer, Game Director, and Lead Artist. The Producer also handled Programming and Sound Design; the Game Director oversaw both Narrative and Game Design; and the Lead Artist was responsible for Animation and Art. Team members seeking guidance or validation could consult the appropriate subteam lead.

Team Dynamics

To avoid overwork and the "crunch" culture prevalent in the video game industry, the team decided to dedicate only three days per week to the project, reserving the rest for other academic obligations. This decision, although it required a reduction in the project scope, was unanimously supported to maintain a healthy working environment. Team members used a shared document to transparently communicate their availability: *"[...] de base, on est contre la philosophie du crunch. Donc, on veut éviter de devoir bosser de manière stupide et de trop sur le TFE. On accepte de faire des sacrifices sur le scope." (Fundamentally, we are against the crunch philosophy. So we want to avoid working stupidly and excessively on the final project. We agree to make sacrifices on the scope.)*

The early departure of the team's programmer highlighted the risks of over-reliance on a single person. As the producer noted: *"[...] on a aussi appris que c'est une erreur de notre part de juste compter sur une seule personne, parce que si la personne quitte, le projet est bloqué." (We also learned that it's a mistake on our part to rely solely on one person, because if they leave, the project is stalled.)* A Game Designer had to take over the programming tasks, learning on the job despite lacking prior experience. Later, the Level Designer's internship during the second academic term required the producer to assume additional responsibilities.

Inconsistencies in work habits and coordination further impacted the team efficiency. One member admitted: "*[...] je n'avais pas une très bonne régularité du travail [...]" (I didn't have very consistent work habits).* The Level Designer also noted issues in documentation practices, which complicated collaboration with artists: *"[...] je construisais les niveaux de design à ma sauce, sans forcément faire de documentation." (I designed levels my own way, without necessarily documenting them.)*

Several members, those who were the sole person in their roles such as programming, reported being overwhelmed due to the breadth of responsibilities. The departure of some members also posed difficulties, resulting in the loss of critical skills. Consequently, each member had to fulfil multiple roles.

Methodology

The team initially attempted to adopt the Scrum framework, influenced by the producer's prior professional experience with Scrum and the introductory courses provided by the school. However,

only the producer and the Game Director had any practical experience with Scrum; the remaining team members' knowledge was limited to the basic concepts introduced in their coursework.

Findings from the focus group indicated that the team lacked a theoretical understanding of Agile principles, as evidenced by the absence of mentioning "Scrum" during the focus group, compared to the producer's interview. The producer also realised that his theoretical understanding of Agile and Scrum differed significantly from that of the rest of the team. Moreover, he acknowledged that he did not have the time to provide adequate instruction on Scrum theory and practice. As he explained: *"Parce que pour quelqu'un qui n'a jamais vu, c'est quoi du agile et qui est toujours resté dans le milieu scolaire, académique, c'est très dur d'une manière ou d'une autre d'imposer une autre technique auxquelles ils ne sont pas habitués" (For someone who has never encountered Agile and has always remained in an academic environment, it is very difficult to impose a method they are not used to)*. Additionally, the team lacked the self-discipline necessary for effective implementation.

Rather than attempting to instill theoretical knowledge of Scrum and Agile methodologies, the team, with the help of their producer and academic faculty, opted to simplify their workflow. By adopting a strategy of continuous improvement, they introduced new practices only when necessary. Their workflow was based on a prioritised to-do list with deadlines, which gradually evolved to incorporate time estimation techniques such as "t-shirt sizing", a common Agile practice. Although they did not strictly adhere to Scrum, their focus on iteration and adaptability preserved key Agile principles. This approach proved more accessible for the team than a full-scale Scrum implementation.

The team conducted two types of meetings: general and specialised. General meetings, involving all team members, were initially held weekly to communicate project updates. Over time, they marked the beginning and end of development sprints, aligning more closely with Agile processes. Although the frequency of these meetings shifted to bi-weekly at one point, they reverted to weekly due to communication inefficiencies. The team refined these meetings to be concise and outcome-focused, facilitating faster decision-making.

Specialised meetings focused on specific disciplines such as Game Design, Art, or Programming. Each day was allocated to a different discipline, and only relevant members participated. The Producer and Game Director attended all specialised meetings and relayed key points to the team.

Communications

The team faced significant communication challenges. Several members reported difficulties in effectively communicating with teammates, particularly when encountering problems. One Level Designer acknowledged that a lack of communication hindered progress due to irregular updates and reluctance to report issues: *"Avant, je restais beaucoup dans mon coin et quand il y avait des problèmes, je n'en parlais pas nécessairement, parce que je n'étais pas à l'aise ou que je ne savais pas comment expliquer en quoi ça bloquait" (Before, I kept to myself a lot and when there were problems, I didn't necessarily talk about them, either because I wasn't comfortable or didn't know how to explain what was blocking me).*

These issues were gradually resolved through open discussion. Once team members recognised the problems, they collectively sought solutions.

Meeting efficiency was also a concern. Although meetings were frequent, they often lacked structure and productivity, resulting in extended durations with limited outcomes. The introduction of communication guidelines significantly improved the situation. For example, the team reduced the time spent in general meetings by shifting detailed discussions to specialised meetings. General meetings were then focused on summarising the information discussed during the specialised sessions. This change was highlighted during the focus group:

[...] on a fait des réunions qui étaient plus concises parce qu'avant les réunions globales étaient un peu en mode 'forum' [...] c'était même du répétitif par rapport à ce qu'on disait en réunion de pôle, [...] du coup, on est repassés à une réunion par semaine globale, mais, qui était beaucoup plus synthétique qui était plus en mode, on résume.

([...] we made the meetings more concise. This is because before, the general meetings were kind of like an open forum [...] it was even repetitive compared to what we said in the specialised meetings, [...] so we went back to one general meeting per week, but it was much more concise, more of a summary.)

The team noted that general meetings had previously lacked conciseness and often repeated content already covered in specialised meetings, where everyone could speak freely. This has been significantly reduced; instead of unstructured discussions, team members now provide a weekly summary of their work, reserving detailed exchanges for the specialised meetings.

Another crucial improvement was the systematic communication of individual availability. This helped the team manage who could contribute to the project at different times, an essential adjustment in an academic context where schedules and external commitments varied widely.

Academic Constraints

Balancing schoolwork with project obligations was a continuous challenge. The team had to schedule meetings during evenings and weekends. Additionally, most students prioritised their coursework, making it difficult to focus fully on the project.

Personal commitments, such as part-time jobs, further limited the time available. As a result, the team strategically reduced the project scope to fit the time constraints. One participant from the focus group explained:

Ben, c'était compliqué dans le sens où comme il y avait des rendus académiques sur le côté, autre que de TFE, ben forcément, tu ne te concentres pas à fond sur ça [...]. On a vraiment commencé à mettre en place des bonnes solutions ces derniers mois, [...] parce qu'on a eu plus de temps pour se rendre compte de ces problèmes.

(Well, it was complicated because there were other academic submissions alongside the thesis, so naturally, you can't fully concentrate on it [...]. We really only started to implement good solutions in the last few months, [...] because we finally had more time to recognise these problems).

School Support

Students sometimes perceived the support they received as more subjective than constructive. They expressed a need for feedback that is better aligned with their specific project requirements. One participant stated, *"[...] l'intervention des professeurs n'était pas forcément pertinente. C'était surtout, si on contactait parce qu'on avait tel problème, on avait surtout un retour plutôt subjectif que quelque chose qui va permettre au groupe de continuer de manière pertinente." (The professors' intervention was not necessarily relevant. Often, when we reached out with a problem, the feedback was more subjective than something that would help the group progress effectively).*

This highlights their desire for practical guidance rather than personal opinions.

The interview also revealed a perception that the freedom granted to students, particularly regarding group formation, may have been more harmful than beneficial. One student mentioned that excessive freedom in forming groups had a negative impact.

Nonetheless, school support was not entirely lacking. Several students reported receiving effective guidance from professors, especially concerning the simplification of the methodology.

Furthermore, discussions with professors occasionally led to strategic decisions, such as dropping certain features of the game.

Team B

Team Composition

The team initially comprised 11 members but fluctuated between 9 and 13 throughout the project, eventually stabilising at 9 members, including an external contributor responsible for music. The team's roles were diverse, spanning disciplines such as game design, programming, sound design, art, and technical art. These roles were dynamic, evolving and adapting to the project's changing needs.

The producer role experienced considerable turnover. Initially, the roles of game director and producer overlapped; later, clearer distinctions were made, and the producer role was reassigned. The new producer subsequently left the project, leading to another reassignment. A few months later, this second producer also left.

Currently, producer responsibilities are officially divided between two individuals. This division arose due to burnout and exhaustion, as it was challenging for members to manage their original roles (e.g. game designer or programmer) alongside the demanding producer responsibilities. One producer now focuses primarily on task management, meeting coordination, workflow, and processes, while the other handles technical aspects, maintaining communication tools.

Team Dynamics

Maintaining a positive team dynamic proved particularly challenging for this team. A major issue was the difficulty in preserving the team consistency, as reflected by the departure of seven members: *"Il y a eu sept personnes qui sont parties depuis le départ." (Seven people have left since the beginning).* These frequent departures likely disrupted team cohesion.

While some team members were able to self-organise effectively, others were not, leading to a fragmented approach to project management. Early attempts to unify communication and task management tools proved unsuccessful, with information dispersed across platforms such as Notion, Miro, Google Sheets, and Discord. As a result, organisational chaos persisted outside formal meetings until late in the project.

Efforts to impose a project structure were met with resistance. One participant observed, *"dès lors qu'on a essayé d'imposer une structure ou quoi à la team, il y en avait forcément quelques-uns [...] qui ne l'appliquaient absolument pas" ("as soon as we tried to impose any kind of structure on the team, there were always a few people [...] who would not follow it at all").*

There was also a notable imbalance in team commitment: while some members were highly engaged, others were significantly less involved, creating tension and dissatisfaction among the more active participants.

To address these challenges, particularly burnout and turnover, the team created a role focused on human resources functions to promote well-being. This included administering regular well-being questionnaires: *"[...] suite à tous ces problèmes de départ, de burnout, etc. On a quand même un des membres qui s'étaient occupés de sections RH qui, chaque mois, proposent un questionnaire de bien-être." ([...] following these issues of departures and burnout, one member took charge of HR tasks and distributed a monthly well-being questionnaire)*. However, no specific reference was made to any measures implemented as a result of these questionnaires, and one member expressed uncertainty regarding their purpose.

Methodology

Initially, no formal structure was established, resulting in naturally evolving roles and workflows within the team. Although no official structure was agreed upon, an informal system emerged organically. Weekly global team meetings set a one-week iteration cycle, while smaller, specialised meetings were organised as needed. However, the absence of formal organisation led to fragmented communication and inconsistent information sharing, particularly regarding tool usage. Team members worked on tasks without a clear, consistent structure.

The meetings were reportedly long, sometimes lasting three to four hours. As one participant explained, *“Ce n’était pas structuré [...]. C'était la réunion où les gens disaient ce qu'ils faisaient la semaine. Donc, c'était le sujet de la réunion. [...]. C’était d'ailleurs des réunions… Ça, c'était un des soucis, qui étaient relativement longues.” (It was unstructured [...]. The meeting consisted of people explaining what they had done during the week. That was the meeting’s purpose [...]. These meetings were one of the problems because they were quite long).*

The team also attempted to implement Agile methodologies, such as Scrum, but these efforts were unsuccessful due to misapplication and limited practical knowledge. Successive producers were reluctant to invest time in learning and teaching these methodologies.

Over time, the team adapted by establishing clearer rules for task and time management. These adaptations reflected the team’s composition: *“le projet a été pensé dès le début, sur la team qu'on avait.” (The project was designed from the start based on the team we had)*. The methodology evolved to consider team experience, unify tools, and recognise that some disciplines required more structure than others.

Additionally, the team emphasised simplifying procedures and tools to maintain engagement and clarity. Complex tools and methods, such as complex spreadsheets of tasks, were deemed ineffective.

Communications

The team initially faced significant communication challenges, partly due to the fragmented use of multiple tools such as Notion, Discord, Miro, and Google Sheets. This scattered documentation made it difficult for members to find accurate information, leading to communication breakdowns and inefficiencies.

An interesting insight from the interviews suggests that the team appeared to confuse tools with methodology. They reported attempting to implement new systems, which in practice amounted to adopting new software or modifying existing tools, but they did not mention specific changes in practice to go with these new systems.

Meetings were initially long and inefficient, sometimes lasting three to four hours, reflecting the lack of clear structure. Communication gaps also extended to problem reporting and resolution, as some members hesitated to express concerns: *“Les autres, je leur disais, ça ne vous plaît pas, il n'y a pas de problème, dites-moi pourquoi… Jamais eu une réponse.” (I told others, if you don’t like something, no problem, just tell me why… Never got an answer).* Despite these challenges, subgroups such as Game Designers and programmers maintained good communication throughout the project.

Recognising these issues, the team implemented structured weekly meetings with clearer procedures to review and assign tasks. They also simplified the tools in use, reducing complexity to facilitate clearer communication, settling on Google Sheets for task management and Miro for the roadmap, high project overview, knowledge-sharing…

Crucially, the team realised that effective communication depends more on understanding team dynamics and individual preferences than on tools alone. This statement corresponds to one of the Agile principles of *“Individuals and interactions over processes and tools”*.

Academic constraints

The interviews revealed that burnout was a significant issue, primarily caused by the combined pressures of coursework and project demands. The heavy workload and the need to balance classes with project responsibilities generated stress and exhaustion, negatively impacting both productivity and team cohesion. This burnout was a key factor behind the frequent turnover in the producer role.

Additionally, academic obligations contributed to team tensions due to uneven class participation; only a small, consistent subset of members regularly attended classes. Although not directly related to the project, this imbalance affected overall team dynamics.

School Support

The team found the theoretical courses on production and project management provided by the school to be generally good, with Agile and Scrum methodologies introduced on multiple occasions. However, they noted a lack of practical application and exposure to real-world scenarios: *“Les réalités du terrain, je pense, n'ont pas été assez montrées à l'école” (“I believe the realities of the field were not sufficiently demonstrated at school”)*.

As a result, students had to rely heavily on their own initiative and experimentation to establish working project management strategies. Rather than following structured methodologies provided by the school, they often adapted or created tools through a process of trial and error. This approach involved a steep learning curve and required significant effort: *“Donc dans les faits, effectivement, on s’est débrouillé tout seul. Et, c'est pour ça que ça a pris du temps” (“So, in practice, we really managed on our own. That is why it took time”)*.

Personalised follow-ups and feedback sessions were appreciated and considered helpful, offering constructive feedback and support. However, they criticised the timing of these sessions, suggesting that they would have been more effective if introduced earlier in the process: *“Les suivis ont été pas mal [...] par contre, c’est arrivé tard” (“The follow-ups were good [...] but they came too late”)*. They indicated that they would have preferred targeted courses or follow-up sessions on project methodology to be offered during the first term of their first year, as well as at the beginning of the second year.

Team C

Team Composition

The team consisted of eight members, including a game director, a producer, game artists, a technical artist, game designers, a sound designer, a programmer, and a level designer.

Team members often took on multiple roles, requiring flexibility in task management, as no member could dedicate themselves full-time to a single discipline: *“[...] on a tous dû prendre beaucoup de casquettes, en tout cas plusieurs casquettes. Donc forcément, ça, dans l'organisation, il faut le prendre en compte [...]” ([...] we all had to wear many hats, in any case several hats. So, naturally, this must be taken into account in the organisation [...]).*

In the beginning, role overlap was common; multiple team members frequently worked on the same tasks in game design or art, reflecting insufficient role demarcation: “On pouvait se trouver à trois dans une tâche de game design ou ce genre de choses. Pour les artistes pareil.” (We could find ourselves three people working on one game design task or similar. It was the same for the artists). This problem was resolved once the team defined specific responsibilities more precisely.

Team Dynamics

Team dynamics were affected by the departure of some members, which required redistributing their tasks among those who remained. This disrupted the workflow and forced the team to adapt quickly. One participant explained, *“[...] on a dû s'adapter à son départ. Du coup on a dû un petit peu re-dispatcher le travail, etc.” (“[...] we had to adapt to their departure. So we had to slightly redistribute the work, etc.”)*.

The team producer played a key role in teaching agile practices, including leading a detailed course on agile and Scrum frameworks. This foundation helped organise the team’s workflow and was well received by members: “*[...] la méthodologie a été beaucoup structuré par le producteur, qui s'est donné un mal de chien pour mettre ça en place, et qui a très vite fonctionné, qu'on a très vite pris au jeu” (“[...] the methodology was largely structured by the producer, who worked extremely hard to implement it, and it worked quickly; we quickly embraced it”).*

The team set guidelines to respect each member’s availability and work-life balance, establishing expectations for responsiveness during regular working hours:

> *Donc on essaie de se mettre pas un planning, mais plutôt des heures où on sait qu'on devrait avoir une réponse de la personne. Voilà, des heures de disponibilité, des heures où on sait qu'on devrait avoir une réponse de la personne.*
>
> *(So we try not to set a strict schedule, but rather hours when we know we should get a response from the person. Availability hours, times when we expect a reply).*

There was a clear preference for in-person meetings over remote ones, as face-to-face interaction improved communication and made meetings more enjoyable. This preference reflects the difficulties of maintaining effective team dynamics in virtual settings: *“C'est mieux de faire ça en présentiel, c'est toujours plus agréable.” (“It’s better to do this in person; it’s always more pleasant.”).*

Methodology

The team initially adopted Scrum as their project development framework, but the sustained frequency of school work and class required adaptation, especially regarding the schedule of meetings. The choice of Scrum was influenced by the producer, whose professional experience in team management inspired trust among members.

Not all events of Scrum were fully used in the beginning to avoid overwhelming team members with too much information. The producer initially introduced only daily meetings and retrospectives, gradually incorporating other events over time. Daily meetings were prioritised to help the team acclimate to the practice early on: *“je ne voulais pas les surcharger d'informations, je voulais vraiment qu'ils s'habituent déjà avec le système de faire un daily” ("I didn’t want to overload them with information; I really wanted them to first get used to the system of doing a daily meeting")*. Retrospectives were also introduced early, as the producer considered them the most important meeting within the Scrum framework. So he introduced the most important events first.

Daily Scrum meetings were conducted asynchronously on a text channel within their chat software (Discord), primarily due to time availability related to the courses. Each team member posted daily updates detailing what they did, and their plan for the next day.

Sprint planning and reviews took place each Sprint. The retrospective meeting was considered as the most important meeting by the team and was held whenever possible at the end of each Sprint. All of these meetings were sometimes postponed, due to sudden change in the academic schedule.

Task management was challenging for them in the beginning because backlog item goals were imprecisely written and the estimation of each lacked precision. However, the creation of backlog items improved over time because they are now created by the producer or the game director, and each discipline lead is required to write specific detail related to the task. The team does not use time estimation for tasks. Instead, they log the actual time spent on each task using a timesheet. The producer then uses this data to estimate the duration of similar future tasks.

They previously attempted to estimate tasks, but the team found it unhelpful, as they tended to give the same estimates regardless of the task, and the team thought it was a waste of time. Although the producer found time estimation useful, the team ultimately preferred logging actual time, which still provides valuable information for project planning to the producer, as a compromise.

During exams, the team temporarily switched to the Kanban framework, allowing a more flexible workflow without frequent meetings, which helped them iterate quickly to meet the exam deadline.

The producer stated that Scrum was no longer suitable at that stage of the project, as it did not allow the team to maintain the tight pace required. They needed a framework that supported "just-in-time" work without the obligation to organise and attend biweekly meetings:

> *[...] on a dû switcher de Scrum à Kanban, parce que tout simplement, Scrum ne nous permettait pas de tenir le rythme. On avait besoin de quelque chose de beaucoup plus flexible, où on allait pouvoir vraiment travailler en flux tendu, sans avoir à faire des réunions toutes les deux semaines*

> *([...] we had to switch from Scrum to Kanban simply because Scrum didn't allow us to keep up the pace. We needed something much more flexible, where we could truly work in a continuous flow without having to hold meetings every two weeks).*

They switched back to Scrum at the end of the exams because it helped them maintain focus through Sprint goals, which was not the case with Kanban.

Communications

The team reported challenges in sharing new ideas and project-specific information. At times, important details were discussed orally among a few team members but were not documented in the team's knowledge base, leading to fragmented communication.

These issues were particularly noticeable when the team briefly used the Kanban methodology. During this time, members worked in isolation, communicating only within their own disciplines and focusing narrowly on individual tasks: *"[...] quand on utilisait Kanban, il y avait des choses qu'on communiquait simplement pas, [...] on était un petit peu, pas dans notre grotte, mais en tout cas on était entre pôles" ("[...] when we used Kanban, there were things we simply didn't communicate [...] we were not exactly in our own caves, but we stayed within our own areas")*.

Over time, the team recognised the need for better documentation and more structured communication channels. During sprints, their main point of contact was through written daily updates, but these were sometimes incomplete or lacking critical information. To address this, the team created a new *update* channel within their chat software. Members were required to post major project developments, including screenshots and descriptions, allowing everyone to stay informed about ongoing progress in detail.

Oral communication also posed problems when decisions were made informally by a few members but not recorded. To solve this, the team adopted a rule that all decisions had to be documented: *"[...] dans notre logique, c'est ce qui n'est pas écrit n'est pas acté." ("in our logic, what is not written is not considered official").*

Academic Constraints

One of the most significant challenges arising from academic constraints was the organisation of meetings. Coursework and classes consumed most of the students' time, leaving little opportunity to schedule team meetings. Even when the producer was able to find available time slots, meetings often had to be cancelled or postponed due to changes in class schedules. This added considerable pressure on the producer: *"L'indisponibilité d'arriver à créer des créneaux de réunion, ça c'était vraiment le pire, parce qu'il y a des fois où je créais une réunion, trois semaines à l'avance, puis les cours changeaient, donc il fallait annuler" ("The inability to secure meeting times was really the worst, sometimes I scheduled a meeting three weeks in advance, then classes changed, so it had to be cancelled").*

Traditional daily Scrum meetings were impractical due to the differences in team members' schedule. They were instead conducted asynchronously via text on Discord: *"Le daily, on le fait de manière asynchrone et textuelle [...] Le daily, on le fait de manière asynchrone et textuelle" ("We do the daily Scrum asynchronously and in text format [...] We do the daily Scrum asynchronously and in text format")*.

Balancing schoolwork, class attendance, and project responsibilities was also a major challenge. Students described how this made it difficult to advance through sprints, as their focus had to alternate between academic and project-related tasks. As one participant noted: *"Moi je sais très bien qu'à un moment donné je choisissais les cours auxquels j'allais. Parce que sinon faire toute la méthodologie à côté alors qu'il y avait les cours, il y avait le mémoire..." ("I clearly remember that at one point I had to choose which classes to attend. Because otherwise, managing all the methodology alongside classes and the thesis was just too much").*

School Support

The team positively received the theoretical introductory course on Agile methodologies and more specifically on Scrum provided by the school. However, the producer noted that prior hands-on experience could have further improved the team's understanding of Scrum by providing practical context before the project began.

Ongoing support throughout the academic year was also seen as beneficial. It provided meaningful guidance and relevant feedback during project reviews, particularly due to the expertise of certain faculty members: *"Niveau école, je dirais quand même que Laduron a du bagage là-dedans aussi, et à chaque fois qu'on avait des reviews avec lui, parce que je pense qu'on en avait au début d'année encore, ça nous aidait" ("At the school level, I would say that Laduron has a background in this, and every time we had reviews with him, especially early in the year, it helped us").*

However, the producer pointed out that certain roles, such as that of the producer, can be perceived as difficult and underappreciated. They suggested that the school could do more to raise awareness about the demanding nature of these roles and to check in more frequently with teams to assess how they are managing such demanding responsibilities: *"Je pense que l'école pourrait sensibiliser peut-être un petit peu plus à ces rôles qui sont des rôles super ingrats, qui peuvent être super mal vécus" ("I think the school could raise more awareness about these roles, which are very thankless and can be very difficult to experience").*

Discussion

Choice of Project Management Framework

Each team adopted a distinct approach to project management. Notably, two students in this cohort had several years of professional experience in team management and Agile practices. Consequently, they were naturally appointed by their teammates as project managers, referred to as producers, for Teams A and C. However, neither had prior experience in video game development. Both observed that managing a student team posed unique challenges compared to managing professionals.

Two of the three teams initially attempted to adopt the Scrum framework, driven by two primary factors: the producers' professional backgrounds inspired trust among team members, and an introductory course on Agile and Scrum provided a foundational understanding of these methodologies. Despite this, only one team, Team C, retained Scrum throughout the project. The other team adapted it, citing a lack of training, insufficient in-depth knowledge, and the practical difficulty of adhering to Scrum's rigid structure in a constrained time context. As Majeed (2012) notes, effective Scrum implementation requires significant training and experience, challenges that exist even among professional teams.

While each team described the school's introductory course on Agile as a good starting point, the team still needed to independently acquire much of their knowledge about Agile and Scrum. During interviews, Team C acknowledged their initial confusion with Scrum practices, requiring time to fully understand and implement them. In hindsight, they considered themselves fortunate to have had an experienced team member to guide them.

Team B, by contrast, did not reference Agile or any formal project management methodology, at least in the beginning. Their workflow developed "naturally" over time. Interestingly, their approach reflected core Agile values, particularly the emphasis on "individuals and interactions over processes and tools." None of the teams employed a traditional Waterfall model, which aligned with the school's structure of regular check-ins and playtest, that required teams to frequently deliver game builds and documentation.

Practices Adaptation

Teams A and C both began with Scrum as their development framework. However, due to a lack of training and practical knowledge, only Team C continued to use it. Team A found Scrum too complex to implement effectively in their context and transitioned to a simplified, custom framework. This approach ultimately resembled that of Team B. The difficulty of applying Scrum "by the book" mirrors findings from Masood et al. (2018), where students struggled with Scrum implementation in a university setting. But their study focused exclusively on software engineering projects; video game development introduces additional challenges unique to the medium (Blow, 2004).

While Team C's experience demonstrates that successful adoption of Agile practices is possible, it requires extensive training, personal commitment, and a high level of self-management. Additionally, both Team A and Team C benefited from the presence of a team member with professional experience in Agile methodologies, unlike Team B. This expertise clearly provided an early advantage by offering initial structure and knowledge at the outset of the project, even though this structure evolved over time, which is, in essence, aligned with Agile principles.

For instance, a key adaptation made by Team C involved the Daily Scrum. Also referred to as the Daily Standup, this meeting is meant to be conducted while standing, to ensure it remains brief and focused (Schwaber & Sutherland, 2020). They replaced this with asynchronous daily text

updates via their chat tool, due to scheduling constraints in the academic setting. The producer expressed an intention to return to in-person standups once the team transitions to professional work, recognising that some benefits are lost in the asynchronous format. Nonetheless, this adaptation offers a viable solution for teams working remotely or asynchronously.

Interestingly, Team A also implemented a variant of *daily coordination*. Instead of a full-team daily meeting, they held discipline-specific meetings focused on particular areas of development. The producer and game director attended all of these meetings and acted as information bridges across disciplines. Although this practice diverges significantly from a traditional daily stand-up, it represents another form of contextual adaptation. However, it places considerable demands on the producer and game director to maintain effective communication and knowledge-sharing.

Teams' Perception of Agile

Team B was the only group that did not mention Agile methodologies during their interview. They viewed such approaches as unnecessarily complex for their needs. In contrast, Team C was the only team to embrace and adopt most principles of Scrum. However, even they expressed reluctance to invest significant time in fully mastering the methodology. As one of the producers explained:

> *Pour moi, la méthodologie, ça s'adapte, c'est juste les connaissances que tu dois avoir sur telle ou telle méthode que tu vas adapter à l'outil. Dans ce cas-là, les connaissances qu'on avait sur la méthodologie, on s'y intéressait pas spécialement… Et on a fait comme on a pu.*
>
> *(For me, methodology is something you adapt; it's just a matter of knowing the specific method so you can tailor it to the tool. In this case, we weren't particularly interested in learning the methodology in depth... We just did what we could.)*

For Team A, the unsuccessful attempt to implement Scrum ultimately led to a more suitable approach, which they adapted to fit their specific context. While Scrum might have worked with more extensive training and time, the constraints of the academic environment made such an implementation difficult.

Although not explicitly stated by Team C, it appears that the producer assumed the role of Scrum Master, while the Game Director acted as the Product Owner. This distribution of roles can vary significantly across teams, including in professional settings (McKenzie et al., 2021).

Challenges and Area for Improvement

This section outlines the common challenges faced by student teams during their projects, emphasising how these difficulties relate specifically to video game production within an academic setting. It also highlights areas where improvements could be made, either through institutional support or structural changes.

Schedule Constraints

Scheduling constraints emerged as a significant and recurring challenge across all teams, regardless of whether they followed a specific Agile framework. Organising regular meetings, crucial for communication and information exchange, proved particularly difficult due to the high number of weekly courses, which left little flexibility outside evenings and weekends. This issue aligns with observations made by Masood et al. (2018), who noted similar difficulties in time management within collaborative academic projects.

The interviews also revealed a broader tension between academic commitments and project responsibilities. Students often felt forced to choose between coursework and the project, as managing both proved to be unsustainable. Team C, for instance, reported doing rotating class attendance among members to prioritise project work, a decision that led to criticism from the school. Similarly, some members of Team B admitted to attending classes while mentally disengaged, preferring to focus on their project instead.

To address this conflict, a potential area for improvement would be to reduce the number of scheduled classes during the week or to formally allocate dedicated time within students' timetables for team meetings and project coordination. As an echo to Kropp and Meier (2013) experiments, students stated that being able to work only a couple of hours every week on their project is not ideal. Many suggested an intensive week instead, without needing to balance their time between other academic work.

Member Departures

All teams reported that member departures had a negative impact on their projects, though the timing and severity varied. Each team responded to these losses in different ways, with varying degrees of success.

An imbalance of roles and disciplines was noted across all teams. In one case, following the early departure of the team's sole programmer, another member, who had no prior experience in programming, had to take over and learn on the job. Another team experienced the departure of seven members over the span of 1.5 years. While such turnovers are not uncommon in professional environments, the academic context places an additional burden on the remaining team members, particularly on team leaders. They were forced to constantly adapt to shifting team dynamics, while at the same time, working on their other responsibilities as programmers, designers or artists.

Two of the three teams suggested that more oversight by the institution could help mitigate these issues. Specifically, they recommended that the school implement mechanisms to ensure balanced team compositions, with sufficient coverage of all necessary disciplines. Another proposed solution was to adapt project requirements based on the composition of each team. For example, a team without a dedicated programmer could be assessed with lighter expectations regarding gameplay complexity and adherence to programming best practices. In return, they could be evaluated more rigorously in areas such as art direction or rewarded for innovative game design that does not rely on advanced programming techniques.

Team Communication Issues

All three teams reported experiencing communication-related challenges. These issues are consistent with findings from Masood et al. (2018), and are also commonly encountered in the game development industry, even outside academic settings (McKenzie et al., 2021). Communication breakdowns were particularly problematic when students assumed a baseline of self-discipline and autonomy from their teammates, which was not always met.

Communication problems can come from different things; in one team, the non-unification of tools, such as task management, or documentation caused communication issues. However, common points relate to a lack of self-awareness from team members on the importance of communication, meaning communication of problems as soon as possible to not let them create bigger problems, later.

This challenge could potentially be mitigated through more comprehensive training in Agile methodologies, which are specifically designed to improve communication and collaboration within teams.

The Project Manager' Burden

Two of the teams benefited from having a member with prior professional experience in project management. This advantage was reflected in the interviews: experienced leadership reduced the likelihood of failure and provided guidance for less experienced teammates, who could rely on the project manager's knowledge and structure.

In contrast, the third team did not have this advantage and struggled with organisational issues in the beginning, stemming from their limited knowledge of effective project management practices. Although the school provided some support, the lack of an experienced leader resulted in a longer adaptation period. The team faced challenges in establishing efficient communication strategies, task management systems, and meeting structures. These difficulties were worsened by the constant rotation of the producer role, mainly due to burnout and fatigue.

While the presence of experienced project managers in two teams was more a matter of coincidence than design, this disparity highlights the need for institutions to offer structured support for students assuming leadership roles. In particular, school should raise awareness of the significant emotional and organisational demands placed on student project managers. As one producer from Team C stated: *"Je pense que l'école pourrait sensibiliser peut-être un petit peu plus à ces rôles qui sont des rôles super ingrats, qui peuvent être super mal vécu suivant comment ça se passe dans le groupe." ("I think the school could perhaps do a bit more to raise awareness about these roles, which are often very thankless and can be extremely difficult, depending on how things unfold within the group.")*.

A further issue is the absence of a dedicated academic specialisation for project management. Students in these projects typically come from one of four disciplines, programming, art, technical art, or game design, and are expected to manage projects in addition to their core responsibilities. While creating a separate project management track may not be necessary, institutions could consider reducing the academic workload or role-specific expectations for students who take on project leadership. This would allow them more time to develop an interest in project management and receive tailored instruction. Since project managers play a critical role in facilitating the adoption of effective production structures, they would benefit from more targeted training to support both their own development and the functioning of their teams.

Institutional Support

Overall, students appreciated some aspects of the institutional support provided, particularly the theoretical courses offered on project management. All teams acknowledged that the school introduced key concepts such as Scrum and Agile, which served as a useful starting point. Team C, in particular, found value in the guidance provided by experienced faculty during project reviews, and producer-specific meetings were seen as beneficial spaces for peer exchange.

However, there were several recurring criticisms. A central concern among all teams was the limited practical application of these theoretical lessons. Students often felt unprepared to apply what they had learned in real-world conditions, leading many to rely on experimentation, and trial-and-error approaches. This gap between theory and practice was especially challenging at the beginning of the project, where early structural issues could have been mitigated through more hands-on preparation and timely feedback.

Another common critique was the timing and relevance of institutional interventions. While personalised feedback sessions were considered helpful, they were often delivered too late to influence early project decisions. Additionally, Team A expressed frustration with feedback that felt overly subjective and not actionable enough. There was also concern that the high degree of autonomy granted to students, particularly during teams' formation, sometimes worsened the structural imbalances that could have been addressed early on by the school.

The interviews also revealed that the lack of time allocated to the project in the early stages was a significant issue. Students felt overwhelmed by the demands of coursework and classes, often struggling to reconcile academic responsibilities with the needs of the project.

Also, based on Kropp and Meier (2013) experiments, they think that putting an emphasis on group dynamics is crucial. Students should have access to a room where they can meet for standups and have a wall for their task board.

Lastly, the demanding nature of leadership roles, such as producers or project managers, was highlighted as an area in need of greater institutional awareness. Students suggested that the school should better recognise the burden of these roles and offer targeted support or adjustments to ease the responsibilities placed on student “leaders”.

Learning Outcomes

Each team provided valuable learning outcomes in their interview, reflecting on their experiences over time. Some of these learning outcomes are listed here, in the hope of giving other students to maybe potentially improve their future project and focus on the most important points of project management.

Importance of Communication

All three teams consistently emphasised the critical role of communication. Team C, in particular, underscored that agility is fundamentally built upon communication. Retrospective meetings and Agile methodologies were regarded as mechanisms to ensure consistent and transparent communication throughout the project. As the producer of Team C stated: *“L'agilité, les gens ont tendance à l'oublier, mais la force de l'agilité, c'est la communication. C'est vraiment le point le plus important.” ("Agility, people tend to forget, is rooted in communication. That’s really the most important point.").*

Adapting the Project to the Team

A key insight shared across teams was the importance of adapting project management methodologies to suit the team members and project context, rather than rigidly adhering to predefined frameworks. This flexibility allowed students to better align methods with the specific needs and dynamics of their teams.

Role Distribution and Cross-Functional Awareness

Teams highlighted the importance of avoiding over-reliance on a single individual for any critical role. Imbalances in skills and disciplines are common in a university setting, where students typically cannot recruit based on specific competencies. One team adopted the practice of encouraging members to develop an interest in and basic understanding of roles beyond their own. This cross-functional awareness creates redundancy, which is especially beneficial when a role is filled by only one person.

In-Person Meetings Over Remote Communication

Students expressed a preference for in-person meetings, which they found more effective for communication and collaboration. Physical presence allowed for more immediate feedback, stronger interpersonal connections and greater team cohesion compared to virtual or asynchronous communication.

Role Clarity and Decision-Making Processes

The early clarification of roles and responsibilities was identified as a crucial practice. Teams recommended documenting each member's role and defining clear areas of responsibility from the outset. Furthermore, appointing a designated decision-maker was seen as essential to resolve conflicts and ensure efficient progress. As one participant noted: *“Tout de suite mettre sur papier les rôles et les responsables des différents responsabilités et pôles du projet et avoir quelqu’un qui vient trancher dans les décisions.” ("Immediately document roles and responsibilities for different areas of the project and have someone who can make final decisions.").*

Study Limitations and Area for Improvement

As a student enrolled in the same Master’s program as the participants, I acknowledge the potential for bias in this study. Additionally, certain contextual insights, though not derived directly from the interview data, were occasionally included to enrich the analysis. Efforts were made to minimise these inclusions and to maintain a neutral perspective throughout the study.

The limited sample size, consisting of only three teams from the same school and Master’s program, restricts both the quantity and diversity of the data. Unfortunately, for Team B, I was only able to interview the two co-producers. Due to time and resource constraints, I was unable to conduct a focus group as I did with the other team. As a result, some of the statements made by Team B may not reflect the views of the entire team.

Also, broader insights could be gained through a larger-scale study involving student game development teams from across multiple universities.

Conclusions

Findings indicate that while Agile frameworks such as Scrum offer valuable structure, strict adherence was often impractical in an academic context. Instead, teams developed hybrid approaches, akin to Scrumban practices, but tailored to their needs. The effectiveness of these adaptations depended largely on the team's prior experience, especially the experience of their project manager, communication habits, and dedication.

Communication emerged as a central challenge across all teams. Ineffective practices, whether due to tool fragmentation, asynchronous updates or lack of specific guidelines to communicate new ideas or changes, were consistently linked to the emergence of conflict and project delays. Teams that established transparent, focused communication protocols reported higher levels of coordination and morale.

Academic constraints such as concurrent coursework and inconsistent schedules further complicated development. Students often had to face trade-offs between academic responsibilities and project tasks, leading to a reduction in scope or the need for more flexible workflows. These challenges suggest the need for higher education institutions to reassess project timelines and course loads during major team projects.

School support was found to be well-intentioned but sometimes misaligned with the realities of the students. While theoretical instruction in Agile methodologies was appreciated, students called for improvement. Specific recommendations included the formation of more balanced teams, more practical training on Agile methodologies, and more support towards students undertaking project management roles.

Despite these limitations, the project outcomes demonstrated meaningful learning. Students acquired not only technical and management skills but also a deeper understanding of collaboration, leadership, and adaptability, skills essential in the professional world.

Future research should investigate a broader range of higher education institutions and team projects to better generalise these findings.

Bibliography

Ahmed, T., Cox, J., Girvan, L., Paul, A., Paul, D., & Thompson, P. (2014). *Developing Information Systems: Practical Guidance for IT Professionals* (J. Cadle, Ed.). BCS Learning & Development Ltd.

Alaidaros, H., Omar, M., & Romli, R. (2021). The state of the art of agile kanban method: Challenges and opportunities. *Independent Journal of Management & Production*, *12*(8), 2535–2550. https://doi.org/10.14807/ijmp.v12i8.1482

Alqudah, M., & Razali, R. (2018). An Empirical Study of Scrumban Formation based on the Selection of Scrum and Kanban Practices. *International Journal on Advanced Science, Engineering and Information Technology*, *8*(6), 2315–2322. https://doi.org/10.18517/ijaseit.8.6.6566

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). *Manifesto for Agile Software Development*. https://agilemanifesto.org/

Blow, J. (2004). Game Development: Harden Than You Think: Ten or twenty years ago it was all fun and games. Now it's blood, sweat, and code. *Queue*.

Boehm, B. (2002). Get ready for agile methods, with care. *Computer*, *35*(1), 64–69. https://doi.org/10.1109/2.976920

Boehm, B., & Turner, R. (2003). Using risk to balance agile and plan- driven methods. *Computer*, *36*(6), 57–66. https://doi.org/10.1109/MC.2003.1204376

Brooks, F. P. (1995). *The mythical man-month: Essays on software engineering* (Anniversary ed). Addison-Wesley Pub. Co.

Chandler, H. M. (2020). *The game production toolbox*. CRC Press, Taylor & Francis Group.

Cho, J. J. (2010). *An Exploratory Study on Issues and Challenges of Agile Software Development with Scrum* [Dissertation]. Utah State University.

Coad, P., Lefebvre, E., DeLuca, J., & De Luca, E. (1999). *Java modeling in color with UML: Enterprise components and process ; [enrich the content of your models, become a better modeler by example, deliver frequent, tangible, working results]*. Prentice Hall PTR.

Dimitriadis, G., & Kamberelis, G. (2024). Chapter 41: Focus Groups—Strategic Articulations of Pedagogy, Politics, and Inquiry. In N. K. Denzin, Y. S. Lincoln, M. D. Giardina, & G. S. Cannella (Eds.), *The SAGE handbook of qualitative research* (Sixth edition, pp. 143–162). SAGE.

Dybå, T., Dingsøyr, T., & Moe, N. B. (2014). Agile Project Management. In G. Ruhe & C. Wohlin (Eds.), *Software Project Management in a Changing World* (pp. 277–300). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-55035-5_11

Engström, H., Berg Marklund, B., Backlund, P., & Toftedahl, M. (2018). Game development from a software and creative product perspective: A quantitative literature review approach. *Entertainment Computing*, *27*, 10–22. https://doi.org/10.1016/j.entcom.2018.02.008

Erickson, F. (2024). Chapter 3: A history of qualitative inquiry in social and educational research. In N. K. Denzin, Y. S. Lincoln, M. D. Giardina, & G. S. Cannella (Eds.), *The SAGE handbook of qualitative research* (Sixth edition, pp. 33–59). SAGE.

Esposito, N. (2005). *A Short and Simple Definition of What a Videogame Is*. 7.

Hodgson, D., & Briand, L. (2013). Controlling the uncontrollable: 'Agile' teams and illusions of autonomy in creative work. *Work, Employment and Society*, *27*(2), 308–325. https://doi.org/10.1177/0950017012460315

Holvikivi, J. (2020). Agile Methods and Team Work in IT Education. In A. Tatnall (Ed.), *Encyclopedia of Education and Information Technologies* (pp. 37–43). Springer International Publishing. https://doi.org/10.1007/978-3-030-10576-1

Holzbaur. (2010). Projects as a Method of Training, Teaching and Research Education. In S. Soomro (Ed.), *New Achievements in Technology Education and Development* (p. 28). InTech. https://doi.org/10.5772/9230

Juricek, J. (2014). Agile Project Management Principles. *Lecture Notes on Software Engineering*, 172–175. https://doi.org/10.7763/LNSE.2014.V2.117

Keith, C. (2020). *Agile game development: Iterative, lean, and scaled practices for game development* (2nd ed.). Pearson Education, Inc.

Koutonen, J., & Leppänen, M. (2013). How Are Agile Methods and Practices Deployed in Video Game Development? A Survey into Finnish Game Studios. In H. Baumeister & B. Weber

(Eds.), *Agile Processes in Software Engineering and Extreme Programming* (Vol. 149, pp. 135–149). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-38314-4_10

Kropp, M., & Meier, A. (2013). Teaching agile software development at university level: Values, management, and craftsmanship. *2013 26th International Conference on Software Engineering Education and Training (CSEE&T)*, 179–188. https://doi.org/10.1109/CSEET.2013.6595249

Leffingwell, D. (2007). *Scaling software agility: Best practices for large enterprises*. Addison-Wesley.

Majeed, H. (2012). Issues and Challenges in Scrum Implementation. *International Journal of Scientific & Engineering Research*, *3*(8), 4.

Masood, Z., Hoda, R., & Blincoe, K. (2018). Adapting agile practices in university contexts. *Journal of Systems and Software*, *144*, 501–510. https://doi.org/10.1016/j.jss.2018.07.011

McKenzie, T., Morales-Trujillo, M., Lukosch, S., & Hoermann, S. (2021). Is Agile Not Agile Enough? A Study on How Agile is Applied and Misapplied in the Video Game Development Industry. *2021 IEEE/ACM Joint 15th International Conference on Software and System Processes (ICSSP) and 16th ACM/IEEE International Conference on Global Software Engineering (ICGSE)*, 94–105. https://doi.org/10.1109/ICSSP-ICGSE52873.2021.00019

Oun, M. A., & Bach, C. (2014). *Qualitative Research Method Summary*. *1*(5).

Pałka, B. (2023). Scrum in video games development – Literature review. *Washington Institute of China Studies*, *17*(1), 31–40.

Pimonova, S. (2020). Agile Methodology in Education of IT Students, Application of. In A. Tatnall (Ed.), *Encyclopedia of Education and Information Technologies* (pp. 27–36). Springer International Publishing. https://doi.org/10.1007/978-3-030-10576-1

Poppendieck, M., & Poppendieck, T. (2003). *Lean Software Development: An Agile Toolkit*.

Poppendieck, M., & Poppendieck, T. (2010). *Lean software development: An agile toolkit* (Nachdr.). Addison-Wesley.

Royce, W. W. (1970). MANAGING THE DEVELOPMENT OF LARGE SOFTWARE SYSTEMS. *The Institute of Electrical and Electronics Engineers*, 1–9.

Ruonala, H.-R. (2016). *Agile Game Development: A Systematic Literature Review*.

Schach, S. R. (2004). *An introduction to object-oriented systems analysis and design with UML and the unified process*. Stewart Mattson.

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide*.

Semenchenko, N., Melnychuk, V., Zaporozhchenko, A., & Molchanov, M. (2025). *VIDEO GAME INDUSTRY AS A PART OF CREATIVE AND DIGITAL ECONOMICS*. https://doi.org/10.5281/ZENODO.15298383

Sommerville, I. (2016). *Software engineering* (Tenth edition). Pearson.

Tatnall, A. (Ed.). (2020). *Encyclopedia of Education and Information Technologies*. Springer International Publishing. https://doi.org/10.1007/978-3-030-10576-1

The 17th State of Agile Report (Survey No. 17th; p. 26). (2023). digital.ai. https://digital.ai/resource-center/analyst-reports/state-of-agile-report/

Thomas, D. R. (2006). A General Inductive Approach for Analyzing Qualitative Evaluation Data. *American Journal of Evaluation*, *27*(2), 237–246. https://doi.org/10.1177/1098214005283748

Vacanti, D., & Coleman, J. (2020). *Kanban Guide*. https://kanbanguides.org/english/

Watson, R. T., Kelly, G. G., Galliers, R. D., & Brancheau, J. C. (1997). http://www.jstor.org Key Issues in Information Systems Management: An International Perspective. *Journal of Management Information Systems*, *13*(4), 91–115.

Abstract

Managing a project as a student in a higher education institution presents unique challenges compared to projects conducted in professional contexts. Academic constraints can interfere with the implementation of standard Agile practices such as Scrum or Kanban. Existing research on this topic is limited and does not directly address video game projects. Furthermore, studies on video game production typically focus on professional development studios rather than academic settings. Through interviews with three student-led projects conducted in an academic context, this study found that industry-standard Agile methodologies, which are already challenging to implement in professional environments, become even more complex in academic settings. This complexity arises from limited time and attention that students can dedicate to such projects due to coursework and other academic responsibilities, as well as a lack of experience and training. The student mindset is sometimes misaligned with methodologies that require a professional outlook. Nonetheless, the findings also reveal that higher education institutions can play a significant role in supporting student success. Moreover, students often develop creative strategies to navigate these challenges, acquiring valuable practical knowledge and experience that can later be applied in professional contexts.

Keywords

Project Management, Higher Education, Video Game, Students, Agile