ENHANCING TEAMWORK DURING MOBILE GAME DEVELOPMENT: POTENTIALS FOR THE DESIGN OF COLLABORATION TOOLS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

İZEL ÇELİK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL DESIGN

JANUARY 2025

Approval of the thesis:

ENHANCING TEAMWORK DURING MOBILE GAME DEVELOPMENT: POTENTIALS FOR THE DESIGN OF COLLABORATION TOOLS

submitted by **İZEL ÇELİK** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Industrial Design, Middle East Technical University** by,

Prof. Dr. Naci Emre Altun
Dean, **Graduate School of Natural and Applied Sciences** _____

Prof. Dr. Gülay Hasdoğan
Head of the Department, **Industrial Design** _____

Assist. Prof. Dr. Güzin Şen
Supervisor, **Industrial Design, METU** _____

Examining Committee Members:

Prof. Dr. Hüseyin Hacıhabiboğlu
Modeling and Simulation, METU _____

Assist. Prof. Dr. Güzin Şen
Industrial Design, METU _____

Assist. Prof. Dr. Sedef Süner Pla Cerda
Industrial Design, TEDU _____

Date: 08.01.2025

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Last name : İzel Çelik

Signature :

ABSTRACT

ENHANCING TEAMWORK DURING MOBILE GAME DEVELOPMENT: POTENTIALS FOR THE DESIGN OF COLLABORATION TOOLS

Çelik, İzel
Master of Science, Industrial Design
Supervisor : Assist. Prof. Dr. Güzin Şen

January 2025, 134 pages

In the game development workflow, professionals with different roles and backgrounds get together to collaborate. Within this workflow, several tools and methods are used to enhance collaboration and overcome the communication gaps resulting from the involvement of diverse disciplines in teamwork. This study aims to examine these tools and methods to identify design intervention areas for the improvement and development of collaboration tools for game developers with a particular focus on “hypercasual” game development teams, which are interdisciplinary small teams comprising designers, artists, programmers, and other specialized roles. Therefore, it searches for the answers to the following questions: What content is communicated within the game development team? What are the benefits and shortcomings of the existing tools and/or methods? How can the communication and collaboration within the game development team be supported by design during the game development processes? The thesis responds to these questions with two-partite research consisting of semi-structured interviews and focus groups conducted with game developers. Interviews inquire about developers’

personal perspectives on the subject matter, while focus groups facilitate collaborative discussions among developers. Building on the analysis of both studies, it identifies the themes that will guide the design of new means of collaboration for mobile game development.

Keywords: Collaboration, computer-supported cooperative work, game development, hypercasual games

ÖZ

MOBİL OYUN GELİŞTİRME SIRASINDA EKİP ÇALIŞMASINI GELİŞTİRME: İŞ BİRLİĞİ ARAÇLARI TASARIMI İÇİN POTANSİYELLER

Çelik, İzel
Yüksek Lisans, Endüstri Ürünleri Tasarımı
Tez Yöneticisi: Dr. Öğr. Üyesi Güzin Şen

Ocak 2025, 134 sayfa

Oyun geliştirme sürecinde, farklı roller ve geçmiş deneyimlere sahip profesyoneller bir araya gelerek iş birliği yapar. Bu farklı disiplinlerin ekip çalışmasına dahil olmasıyla birlikte ortaya çıkan iletişim kopukluklarını azaltmak ve iş birliğini geliştirmek için çeşitli araçlar ve yöntemler kullanılır. Bu çalışma, bu araç ve yöntemleri inceleyerek, oyun geliştiricilerin ihtiyaçlarına cevap verecek iş birliği araçlarının geliştirilmesi için tasarım müdahalesi alanlarını belirlemeyi amaçlamakta, bunu yaparken tasarımcılar, sanatçılar, yazılımcılar ve diğer uzmanlaşmış rolleri içeren, küçük ve disiplinler arası "aşırı basit" oyun geliştirme ekiplerine odaklanmaktadır. Bu amaca ulaşmak için sorulan sorular şunlardır: Oyun geliştirme ekibi içinde hangi içerikler iletilmektedir? Mevcut araçların ve/veya yöntemlerin faydaları ve eksiklikleri nelerdir? Oyun geliştirme sürecinde ekip içi iletişim ve iş birliği tasarım yoluyla nasıl desteklenebilir? Bu sorulara yanıt bulmak için oyun geliştiricilerle yapılan yarı-yapılandırılmış mülakatlar ve odak grupları kullanılmıştır. Mülakatlar, geliştiricilerin konuyla ilgili kişisel bakış açılarını ortaya koyarken, odak grupları geliştiriciler arasında iş birliğine dayalı tartışmaları ortaya

çıkarmaktadır. Tez, her iki çalışmanın analizine dayanarak, mobil oyun geliştirme sürecine entegre edilebilecek iş birliği araçlarının tasarımı için kılavuzluk edecek temalar önermektedir.

Anahtar Kelimeler: İş birliği, bilgisayar destekli ortak çalışma, oyun geliştirme, aşırı-basit oyunlar

To the ones who raised me

ACKNOWLEDGMENTS

I would like to express my gratidute to my advisor Assist. Prof. Dr. Güzin Şen for her guidance, continuous support, and sincere encouragements throughout this process. I am forever grateful for her belief in me, without which I couldn't have achieved completing this journey. I would also like to thank my thesis defense committee for their insightful comments and discussions.

I would like to acknowledge my colleages from the Department of Industrial Design at METU for creating a supporting environment and thank them for encouraging me to keep going.

I would like to express my sincere gratitude to my friends, who were with me at my best and worst days through my life, both of which were steps while I completed my journey. In particular, I am deeply grateful for Can Deniz Çam's unwavering support throughout this journey.

I would like to thank my brother, Baran Çelik, for being the best brother a big sister can ask for.

I am deeply appreciative of my extended family, who have always been there for me throughout my academic journey, from kindergarten to the present day.

I would like to thank my mother and my father, Sevginar and Urfan Çelik, for raising me to be who I am today. I can never thank them enough for their care and support. I thank you for being proud of me.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS x

TABLE OF CONTENTS xi

LIST OF TABLES xiv

LIST OF FIGURES xv

1 INTRODUCTION 1

1.1 Aim and Scope of the Research 4

1.2 Research Questions 4

1.3 Structure of the Thesis 5

2 LITERATURE REVIEW 7

2.1 Collaboration 7

2.1.1 Computer-Supported Cooperative Work 8

2.1.2 Collaborative Behaviors 10

2.1.3 Interdisciplinary Collaboration 15

2.2 Game & Game Development 16

2.2.1 Game Definition & Video Games 16

2.2.2 Types & Genres of Games 17

2.2.3 Game Development 21

2.2.4 Hypercasual Game Development 28

2.3 Methods and Tools in Game Development 30

2.4 Collaboration in Game Development: Challenges and Suggestions 37

2.4.1 Game Development After the Pandemic 40

3 METHODOLOGY 43
3.1 Research Approach 43
3.2 Research Stages 44
3.2.1 Sampling 45
3.2.2 Interview Design & Conduct 47
3.2.3 Analysis of Interviews 49
3.3 Generative Workshop 50
3.3.1 Generative Workshop Design and Conduct 50
3.3.2 Analysis of Generative Workshop 54
3.4 Limitations 54
4 FINDINGS & DISCUSSION 57
4.1 Stage 1 Findings 57
4.1.1 Project and Task Management 61
4.1.2 Communication 67
4.1.3 Collaboration 76
4.1.4 Documentation 81
4.1.5 Tool Specific Considerations 86
4.2 Stage 2 Findings 93
4.2.1 Project and Task Management 94
4.2.2 Communication 95
4.2.3 Collaboration 96
4.2.4 Documentation 97
5 CONCLUSION 99
5.1 Overview of the Research 99

5.2 Revisiting Research Questions 100
5.2.1 What content is being communicated between the members of the game design team throughout the (game) development process? 100
5.2.2 What are the strengths and limitations of the existing tools and/or methods utilized in mobile game development for design communication and team collaboration? 103
5.2.3 How can the communication of mobile game design and the collaboration between team members be supported by design during the game development processes? 105
5.3 Future Research 108
5.4 Final Remarks 108
REFERENCES 111
A. Consent Form - Interviews 123
B. Interview Questions (Turkish) 124
C. Interview Questions (English) 126
D. Consent Form – Generative Workshop (Focus Group) 128
E. Toolbox used in Generative Workshop 129
F. Generative Workshop Outcomes 130

LIST OF TABLES

TABLES

Table 2.1 Categorization of Team Behavior (Team Behavior Model) (McGrath, 1991, p. 154) 10
Table 2.2 Categorization of technologies supporting collaborative behaviors and examples (Grudin & Poltrock, 2014) 12
Table 2.3 Roles in Game Development Teams (Adams, 2014; Bates, 2004; Chandler, 2013) 22
Table 2.4 Documentation of Game Design During Development 33
Table 3.1 Participant Information 47
Table 4.1 Strategies and Dimensions of Project and Task Management 60
Table 4.2 Strategies and Dimensions of Communication 67
Table 4.3 Strategies and Dimensions of Collaboration 76
Table 4.4 Strategies and Dimensions of Documentation 81
Table 4.5 Strategies and Dimensions of Tool Specific Considerations 86

LIST OF FIGURES

FIGURES

Figure 2.1. Research and development contexts in CSCW research area defining work levels (Grudin, 1994a, p. 21) 9
Figure 2.2. Examples of tools that facilitate collaboration 14
Figure 2.3. Stardew Valley game on two different platforms: mobile and computer 19
Figure 2.4. Stages of Game Development synthesized by the author 24
Figure 2.5. How Agile practices are applied to different stages of development (McKenzie et al., 2021) 27
Figure 2.6. Screenshots taken from the game “Fidget Trading” developed by MagicLab 29
Figure 2.7. One Page Design: basic format, one page design when it was printed, and one page design after the designer used it during development. (*One Page Design - GDC Presentations*, 2010) 32
Figure 2.8. Three versions of puzzle drafts for the creation of Pyramid Puzzle (Callele et al., 2005, pp. 7-8) 34
Figure 3.1. Social Media invite & Printed Invite 46
Figure 3.2. File organization and initial codes in MaxQDA Software 50
Figure 3.3. Toolbox used in the generative workshop (Appendix E) 53
Figure 3.4. Comic strips created by the participants (Appendix F) 53
Figure 3.5. Groups made by participants (Appendix F) 54
Figure 4.1. Tools used individually by the participants 58
Figure 4.2. Tools used for collaboration 59

CHAPTER 1

INTRODUCTION

For a project to achieve success, proper teamwork is crucial. In the past, teams were composed of people who did the same work; however, as professions get more specialized, the composition of teams has become more varied and multidisciplinary. But how do people with different backgrounds collaborate? To answer this inquiry, the dynamics of multidisciplinary and interdisciplinary work have been elaborated as a significant topic of discussion in design literature. Game development may serve as an ideal context to examine the challenges of collaboration and teamwork in multidisciplinary and interdisciplinary teams.

Teamwork can be defined as the process where the members of a team work together to achieve task goals. It is a broader term that includes both collaboration and cooperation. Collaboration can be defined as the joint effort of working together needed to achieve the shared goal (Driskell et al., 2018), whereas cooperation is used to refer to a way of working together, focusing on the interdependent nature of tasks (Schmidt & Bannon, 1992).

Game development constitutes one domain of design work that is executed as interdisciplinary teamwork, requiring collaboration between diverse roles, including but not limited to programmers, designers, artists, and so on. While game development can be seen as a form of software development, it differs by combining elements of sound, art, and human factors. Engström et al. (2018) separates video games (in this thesis, referred to as "games") from other software products by highlighting the creative product aspect by incorporating art elements. The difference is also reflected in the development process. The game development process has an emphasis on entertainment in addition to functional requirements,

which may result in design changes even in later stages of the development (Kasurinen et al., 2017). Interdisciplinary nature requiring diverse roles in development is another factor that differentiates game development.

The game development process requires a high level of planning due to its multidisciplinary nature. The complexity of diverse roles working together necessitates the implementation of additional activities called articulation work, which includes management, task allocation, and accountability (Schmidt & Bannon, 1992). Combined with the high pressure of market demand and competition (Aleem et al., 2016b), collaboration takes an important role in the process.

Within game development, a variety of tools and methods are utilized to facilitate the design and development process. These tools and methods can be used for the tasks to be completed by an individual, or they can comprise features that allow collaboration and cooperation. Tools utilized by individuals can be employed to generate assets that are delivered to other team members to incorporate into collaborated work, which establishes collaboration. Additionally, tools that facilitate articulation work play a more active and prominent role in enhancing teamwork in game development. Tools and methods used in game development can be examined to identify areas where there lies a need to enhance communication. Enhancing communication may then facilitate the closure of communication gaps that hinder the cooperation and collaboration efforts. Whitson (2018) also considers tools as agents in collaborative development, calling them boundary objects, which makes them paramount for collaboration. Engström (2019) brings attention to research on the development of design tools as an area to be addressed to lessen the gap between game research and industry practices.

One of the reasons that create gaps in communication among the team members involved in game development is the adaptation and experience of different methods for articulation work. Due to its roots in software development, game development processes employ software development strategies like agile methodologies. These strategies serve to organize the development process and they align with the

requirements of articulation work. As this process contains other disciplines in addition to software engineering, the implementation of these methodologies creates challenges, especially relating to design and art, as they are more subjective; it is challenging to quantify and integrate them into a structured framework (Chandler, 2013). Entertainment and emotional response are non-functional requirements of games, and they are difficult to define and describe differently than requirements in traditional software engineering, which makes using methodologies like agile and scrum challenging (Lehtonen et al., 2020). In addition, inappropriate implementation of such software methodologies also adds to challenges that occur in the software development of the game development process (McKenzie et al., 2019).

Documentation-related issues can be considered as another important factor affecting the collaboration and cooperation during game development. Documents like the Game Design Document, in addition to being used for documenting, are used to communicate design ideas and tasks between team members, which require being kept up-to-date, easily accessible, and understood well by all team members. (Bates, 2004; Salazar et al., 2012).

Differences in professional backgrounds and training can lead to difficulty in conveying ideas effectively (Callele et al., 2005), which is another issue faced by game developers during teamwork. The iterative nature of game development requires clear communication between team members to keep the team on the same page (DeWinter & Moeller, 2016). Tran and Biddle (2009) place the background of the team members as a factor that contributes to the success of collaboration; giving an example of how they may present barriers in the form of "biased assumptions (e.g. efficient program code might be regarded as more important than usability in software engineering)" or "issues related to training (certain disciplines favor isolated working conditions)" (Tran & Biddle, 2009, p. 3).

This study will further investigate the aforementioned challenges of collaboration within game development further by focusing on the supporting themes of communication, coordination, and methodologies. As a rapidly growing industry

(Capcom, 2024), game development constitutes an expanding area of interest in research; nevertheless, it is also found to be opaque with its practices that aim to protect development teams or intellectual property (Karabinus & Atherton, 2018). Therefore, it becomes more significant to inquire into this specific domain and shed light on existing practices and to determine the awaiting potentials regarding teamwork in game development.

Recently, a new area of game development has emerged: hypercasual games; their process involves designing and developing small-scope games in a short time span with dynamic, small teams. The quick turnover requires teams to collaborate quickly and smoothly, which can magnify the effect of small challenges encountered during the development process. The recentness of hypercasual games and the nature of the teamwork incorporated in their development process make them an ideal subject to study collaboration, and thus the study focuses on hypercasual game development.

1.1 Aim and Scope of the Research

This research aims to identify areas for design intervention during communication of game design between team members from diverse backgrounds. In this research, the game development process in hypercasual game development is investigated, tools and methods employed by the developers are identified, and practices of the developers are investigated to identify design intervention areas within the game design and development process utilizing tools and methods.

1.2 Research Questions

In this research, the following questions are examined:

> *Q1. What content is being communicated between the members of the game design team throughout the (game) development process?*

Q2. What are the strengths and limitations of the existing tools and/or methods utilized in mobile game development for design communication and team collaboration?

Q.3 How can the communication of mobile game design and the collaboration between team members be supported by design during the game development processes?

1.3 Structure of the Thesis

Chapter 1, Introduction, presents the background, describes the problem, and introduces the aim and research questions of this study.

Chapter 2, Literature Review, reviews the related literature to the topic of the study. Collaboration is examined generally through the frame of computer-supported collaborative work as most tools and methods incorporate computerized technologies in the digital domain of game design. Game definitions, game types, and models identifying the game development process are introduced, followed by studies exploring collaboration specifically within the context of game design.

Chapter 3, Methodology, describes the methodology of the study, inquiring into the qualitative research approach. Research stages of semi-structured interviews and workshops are described, including sampling, conduct, and data analysis processes.

Chapter 4, Findings and Discussion, presents the findings of the study and discusses them in relation to the literature.

Chapter 5, Conclusion, summarizes the study and the conclusions drawn from it by revisiting research questions.

CHAPTER 2

LITERATURE REVIEW

The primary objective of this research study is to investigate collaboration and communication practices among team members with diverse roles involved in the mobile game development process. This chapter examines the literature on collaboration, game, and game development, and the studies conducted on collaboration within the context of game development. The first section examines collaboration and cooperative work, exploring how technology and current collaboration tools apply to it. The second section defines games, game development, the game development pipeline, and roles in game development. The third section looks at research conducted on collaboration during game development.

2.1 Collaboration

This study investigates the challenges associated with collaborative efforts. While there are games developed by a single individual, the majority of games available are the result of multiple people working together. In the Merriam-Webster dictionary, the verb *collaborate* is defined as "to work jointly with others or together especially in an intellectual endeavor" ("Definition of COLLABORATE," 2024). Patel et al. (2012) give a working definition for collaboration; when two or more people are engaged in interaction, working to achieve common goals, within one or more episodes is called collaboration.

In their study, Driskell et al. (2018) define teamwork as working together to achieve goals. Their work focuses on enhancing the effectiveness of teams through the development of teamwork process models by analyzing communication,

coordination, conflict management, and monitoring; however, the significance of tools is not addressed in their work. Another study on the effectiveness of teamwork, which was conducted by Oliveira et al. (2015), investigates the preferences for different mediums of communication and collaboration. Their survey results indicate that face-to-face interactions are preferable to the utilization of digital collaboration tools. However, as the prevalence of remote working increases, the effectiveness of digital tools may need to be reconsidered.

2.1.1 Computer-Supported Cooperative Work

The computer-supported cooperative work research area can be a major source for collaboration studies. To understand this research area, differences between collaboration and cooperation can be discussed. Cooperative work is distinguished from collaboration by its focus; it requires individuals to depend on each other's contributions to accomplish their shared objectives (Schmidt & Bannon, 1992). The need for cooperative work emerges from different needs. These needs are stated by Schmidt and Bannon (1992) as augmentation of human capacities, combining specialized activities, diversifying problem-solving strategies, and application of multiple perspectives.

Cooperative work is significantly influenced by computerized tools, particularly in the domain of game development. Consequently, this section will explore the concept of Computer Supported Cooperative Work (CSCW) and related cooperative behaviors. First coined by Irene Greif and Paul Cashman in 1984, Computer Supported Cooperative Work (CSCW) is a research area where technology and collaborative work intersect (Grudin & Poltrock, 2014). Schmidt and Bannon (1992) classify CSCW as a design-oriented research area, focusing on understanding cooperative work to design solutions that improve cooperation within work contexts.

Characteristics of cooperative work can be listed as: Large and transient units, unstable membership to units, intersecting units, dynamic interaction patterns,

physical and logical distribution of work, out-of-proportion perspectives, and incongruent strategies (Schmidt & Bannon, 1992).

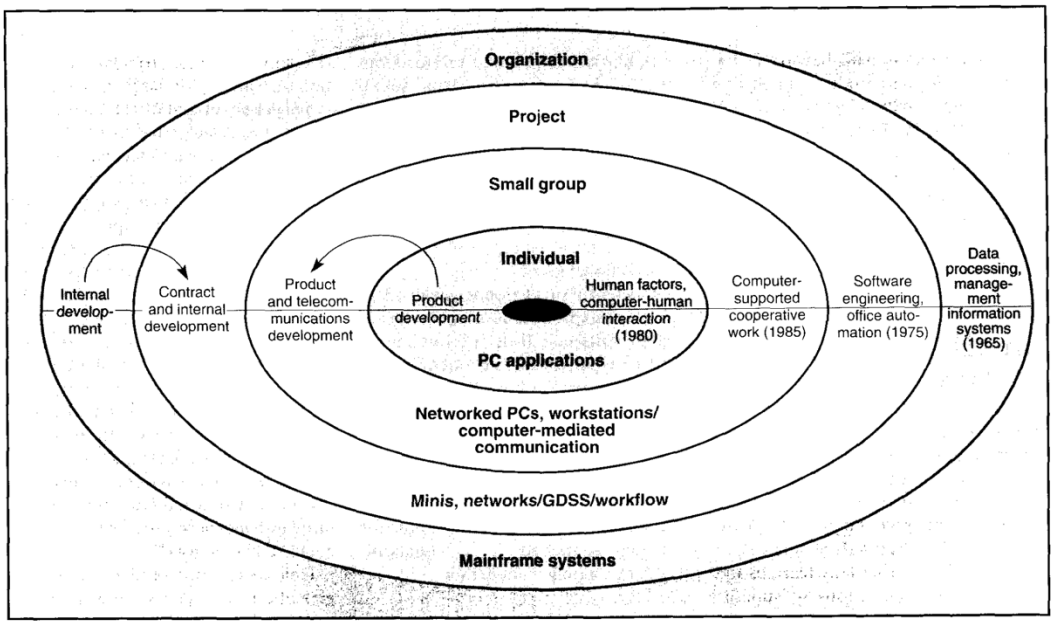


Figure 2.1. Research and development contexts in CSCW research area defining work levels (Grudin, 1994a, p. 21)

CSCW as a research area can be categorized through the scale of organizations it studies. Grudin's (1994a) model categorizes organizations into distinct rings: individual, small group, project, and organization. While Human-Computer Interaction research falls under the individual ring, the CSCW area is placed at the small group level. Systems contained in this level are networked PCs, workstations/computer-mediated communication (Grudin, 1994a). Figure 2.1 illustrates the rings and their respective domains, as well as the research areas that correspond to each ring. Although the focus of CSCW was listed as small groups, today, the area of focus of CSCW expands over the interaction within units of all sizes and how technology is utilized (Grudin & Poltrock, 2014).

2.1.2 Collaborative Behaviors

Collaboration could take many forms; examples of people working together include extreme collaboration, congressional sessions, and policy-making and design. In extreme collaboration, collaborators come together in a room and work together on a complex task while being fully aware of what the others are doing. Congressional sessions involve participants with highly specialized roles engaging in a highly formal manner with established protocols (Ellis et al., 2005). Policymaking and design involve collaborating on intricate, ill-defined problems, known as wicked problems, as introduced by Rittel and Webber (Rittel & Webber, 1973, as cited in Ellis et al., 2005). Game development falls under this final category, characterized by its design-oriented approach and collaborative nature.

Grudin and Poltrock (2014) report three clusters in CSCW research area based on research and development:"(i) the social psychology of groups or teams and technologies that support them; (ii) organizational behavior and support; and (iii) community analysis and support." Especially the first category of studies that deal with team behaviors and supporting technologies with a social-psychological perspective can help us deconstruct teamwork in game development. Within this category, social behavior is analyzed in isolation from organizational context (Grudin & Poltrock, 2014). To illustrate, Joseph McGrath's model (1991) categorizes team behavior under three functions and four modes. Table 2.1 shows behaviors that fall under the categories, which can be used to analyze team behaviors during game development. Production — actions of primary tasks and goals, group well-being — actions of social entity, member support — actions of individual needs and development make up the functions; describing the purposes of actions taken by the teams. Mode dimension categorizes the type of activity (McGrath, 1991).

Table 2.1 Categorization of Team Behavior (Team Behavior Model) (McGrath, 1991, p. 154)

Team Behavior	**Production**	**Group well-being**	**Member support**
Inception (Mode 1)	Production demand and opportunity	Interaction demand and opportunity	Inclusion demand and opportunity
Problem-solving (Mode 2)	Technical problem solving	Role network definition	Position and status achievements
Conflict resolution (Mode 3)	Policy resolution	Power and payoff distribution	Contribution and payoff distribution
Execution (Mode 4)	Performance	Interaction	Principle focus of attention

In a teamwork setting, where intertwined and independent activities occur together, coordination of the work increases complexity as the number of people is greater than a few (Carstensen & Schmidt, 2000). In the roots of this complexity lies the nature of cooperative work, which requires additional activities to manage collaboration. As described in "Taking CSCW Seriously" (Schmidt & Bannon, 1992), additional tasks include management and coordination, which consists of task allocation, accountability, and articulation work. These extraneous activities directly contribute to the development of the final product. Articulation work involves coordinating tasks, efforts, and actors to achieve a common goal within a cooperative work setting (Schmidt & Bannon, 1992). Given these attributes, articulation work is found to be crucial in managing the complexity of group work (Lee & Paine, 2015). Collaboration tools, exemplified in Figure 2.2, support articulation work (Schmidt & Bannon, 1992). It is important to understand what makes the work complex to collaborate on to suggest design guidelines to support this complexity (Carstensen & Schmidt, 2000).

Collaborative behaviors examined under CSCW can be expressed using two primary dimensions: content (communication, information sharing, and coordination) and time (synchronous and asynchronous) (Grudin & Poltrock, 2014). An exemplary categorization of such behaviours is featured in Grudin and Poltrock's work, which

is given in Table 2.2. Similarly, the tools that support these behaviours are commonly used during game development, thus the classification examined under CSCW is relevant to this study.

Table 2.2 Categorization of technologies supporting collaborative behaviors and examples (Grudin & Poltrock, 2014)

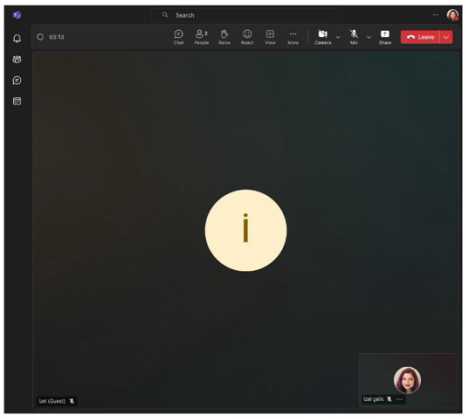
	Communication	**Information Sharing**	**Coordination**
Synchronous	Telephone Video conferencing Instant messaging Texting	Whiteboards Application sharing Meeting facilitation Virtual worlds	Floor control Session management Location tracking
Asynchronous	Email Voice mail Blogs Social networking sites	Document repositories Wikis Web sites Team workspaces	Workflow management CASE tools Project management Calendar scheduling

In 2005, Ellis et al. highlighted the disparity between synchronous and asynchronous environments, which are created by tools that provide only a single mode of operation. Notably, they also call attention to the existence of tools that facilitate the integration of synchronous and asynchronous worlds (Ellis et al., 2005). Popular collaboration tools of today hybridize synchronous and asynchronous features even further; as an example, whiteboard applications (information sharing – synchronous) like Miro are used to refer back to information as with asynchronous information sharing tools. On the other hand, tools like Microsoft Teams highlight their video conferencing features (communication – synchronous) in addition to being team workspaces (information sharing – asynchronous) to collect and store information. Figure 2.2 shows examples of tools that incorporate features like video conferencing/synchronous communication (Microsoft Teams), instant messaging/synchronous communication (Slack), whiteboards/synchronous

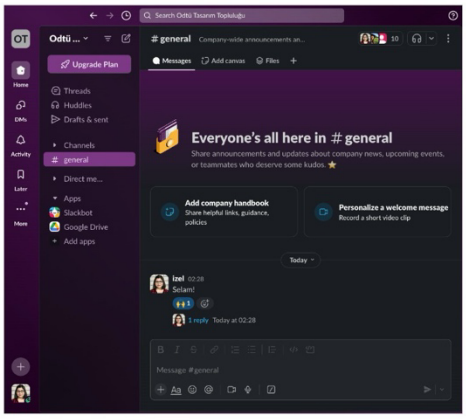
information sharing (Miro), document repositories/asynchronous information sharing (Confluence), and project management tools/asynchronous coordination (ClickUp).

With the emergence of social media networks, multiplayer games, and similar examples, the success of collaboration applications has increased. Unlike collaboration tools of the past, today's tools usually are used on browsers or with the help of plug-ins; with the addition of mobile applications (Olson & Olson, 2012).

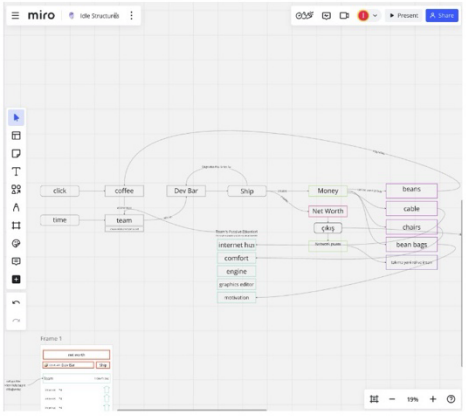
Studies on collaboration tools in the CSCW research area are categorized in Olson and Olson's work (2012) under five titles: technical infrastructure, coordination support, information repositories, social computing, and integrated systems. These categories can be used to analyze and define collaboration tools used in game development. The technical infrastructure is broadly related to tools used in communication, such as the Internet or teleconference technologies. Studies on meeting support systems, workflow tools, group calendars, and awareness make up the coordination support category (Olson & Olson, 2012). Repositories of shared knowledge, wikis, capture, and replay fall under the information repositories, which are also asynchronous information-sharing tools (Grudin & Poltrock, 2014; Olson & Olson, 2012). Social computing lists works on social and psychological dimensions of collaboration. Integrated systems give examples of collaboration systems integrated into workspaces (Olson & Olson, 2012).



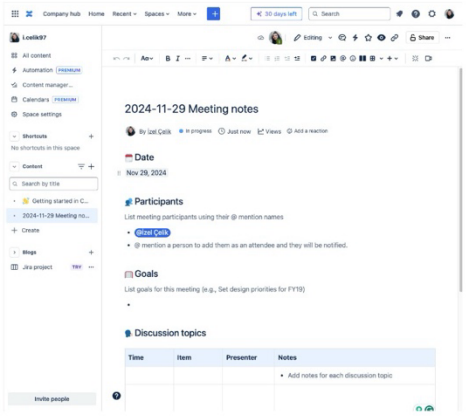
Microsoft Teams



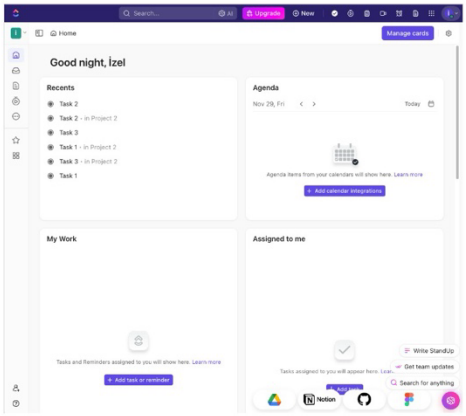
Slack



Miro



Confluence



ClickUp

Figure 2.2. Examples of tools that facilitate collaboration

In 1994, Grudin studied issues which developers should be concerned about while developing collaboration applications; however, not all issues they have pointed out are applicable in today's technology setting (Grudin, 1994b; Olson & Olson, 2012). Most failures of audio-conferencing tools usually connect to poor audio quality (Olson & Olson, 2012). To overcome this, modern communication tools usually employ features that improve the sound quality. Examples like Zoom and Discord, which are communication tools, include built-in noise-cancelling features.

Previous paragraphs explored the primary categories that define cooperative behavior and the tools that support them. Within the realm of CSCW research, the level of "awareness" exhibited by collaborators while utilizing groupware emerges as another topic of keen interest, as teamwork demands each team member's focus on the collaborative efforts of their fellow team members. Awareness as a concept in CSCW emerged between 1992 and 1995, as network technologies were developed to support synchronous collaboration (Grudin & Poltrock, 2014). The term refers to being aware of others' activities, context, and interactions in a shared work setting. Work visibility as a kind of awareness within work can be categorized under three titles: visibility of past work, visibility of current work, and visibility of future work (Ellis et al., 2005). Visibility of work on collaboration creates shared understanding, reduced ambiguity, and improved communication (Huang et al., 2024), and various tools can enhance work visibility.

It is also important to establish an understanding of awareness in CSCW systems (Carstensen & Schmidt, 2000). This also makes creating awareness a goal in supporting collaborative work.

2.1.3 Interdisciplinary Collaboration

While introducing the background of this thesis, the interdisciplinary nature of game development is presented as an aspect that may create issues in collaboration. Interdisciplinarity as defined in academic studies is "bringing together in some

fashion distinctive components of two or more disciplines" (Nissani, 1995, p. 122). On the other hand, within the context of teamwork, interdisciplinary collaboration can be defined as "joint problem solving across disciplines towards a shared goal" (Chung et al., 2022, p. 401). In game development, interdisciplinary collaboration involves combining the knowledge and methods of individuals from diverse fields of art, design, and software engineering, among others, to create a game (Callele et al., 2005).

2.2 Game & Game Development

To better understand the concepts related to collaboration and game development, basic definitions of game development are described in this section.

2.2.1 Game Definition & Video Games

"Game" is defined differently by many researchers; however, most of these definitions describe games as a form of play. Adams' description depicts games as play activities that are acted out within a rule set to achieve a goal or goals (Adams, 2014). The International Game Developers Association (IGDA) gives a definition that is a "working definition": "A game is an activity with rules. It is a form of play often but not always involving conflict, either with other players, with the game system itself, or with randomness/fate/luck." (IGDA Education Special Interest Group, 2008, p. 5) Adams (2014) discusses video games as a subset of games, which is mediated by a computer; in video games, the computer knows and implements the rules. IGDA (2008) approaches differently; they define video games as games that use digital screens. In this thesis, the term "games" is employed to specifically refer to video games that are played through computerized means, particularly mobile devices such as smartphones and tablets.

Within the context of software development, games are defined as "software applications designed to entertain users" (Mizutani et al., 2021). They are

differentiated from software applications by their aim of entertaining the user, rather than solving a problem or enhancing a process (Kasurinen, 2016).

2.2.2 Types & Genres of Games

Games can be categorized in several ways, including based on audience, genre, platform, game modes, and business models.

2.2.2.1 Types

Games can be categorized into three types based on the audience: core games, casual games, and serious games. Core games or hardcore games get their name from their audience, stereotypical hardcore players who invest more time in games that have more complexity (Adams, 2014; Aktaş & Orçun, 2016; Juul, 2012).

The emergence of social media and smartphones has expanded the audience beyond the traditional gaming sphere (Keogh, 2015). Unlike the name suggests, casual players are not players that are less committed, but more flexible; such players may be playing just for a few brief moments, or up to 9 two-hour playing sessions a week (Juul, 2012; Keogh, 2015).

In addition to the properties of games mentioned in its definition, serious games also consist of an educational component (Tran & Biddle, 2009). These types of games are created with a teaching goal at its core. Serious games can also be designed to simulate real-world processes, convey a specific message, or promote health and well-being (Adams, 2014). Serious games provide a valuable domain of study in game research, particularly given the opaque practices of the industry (Karabinus & Atherton, 2018). Consequently, studies on serious games play a significant role in game development research.

2.2.2.2 Genres

In the literature, genre is often utilized while categorizing games. The genre of a game is determined by the type of gameplay challenge it presents (Chandler, 2013), and the genre of the game can affect the development process (Aleem et al., 2016b). Classic game genres are listed in Adams' work: shooter, action and arcade, platform, fighting, strategy, role-playing, sports, vehicle simulation, construction and simulation, adventure, and puzzle (Adams, 2014). However, these genres are not particularly descriptive when discussing hypercasual games due to their limited variance in gameplay.

2.2.2.3 Platforms

Games can also be categorized based on the platforms they are available on. Platforms refer to the devices that games are played on; Adams (2014) categorizes such devices into three categories: home game consoles, personal computers, and portable devices. PlayStation and Xbox are examples of home consoles. Portable devices include smartphones and handheld consoles like Nintendo Switch and PS Vita. Arcade machines, ride simulators, gambling machines, and games built into bus seats are other kinds of devices that games are seen on; however, they are niche platforms that do not get much attention while talking about game design and development (Adams, 2014).

Each gaming platform possesses distinct performance characteristics, input methods, and configurations while gaming (Adams, 2014). For input, home consoles and computers can utilize gamepads; mobile phones utilize touch screens. While mobile games are designed for brief gameplay sessions distributed throughout the day, computer games necessitate a longer focus for each session (Chandler, 2013).

While some games are published on a single platform, multi-platform games are also present and require design changes to fit each platform (Chandler, 2013, Politowski et al., 2021). Examples show control buttons placed on the screen for games that are

ported from consoles to mobile games. Platform constraints are also listed to be issues encountered during game development by developers (Politowski et al., 2021).

Figure 2.3 shows the game Stardew Valley, developed by ConcernedApe (Eric Barone) (Stardew Valley - FAQ, 2024). The mobile version of the game places user interface elements on the sides to leave more space for the gameplay area, in addition to displaying control buttons.



Figure 2.3. Stardew Valley game on two different platforms: mobile and computer, developed by ConcernedApe (Eric Barone)

2.2.2.4 Modes

Games can be categorized based on competition and cooperation: two-player competitive, multiplayer competitive, multiplayer cooperative, team-based, single-player, and hybrid competition modes (Adams, 2014). Most hypercasual games are single-player; however, leaderboards are common, which is a way of implementing competition (Pizzo, 2023).

2.2.2.5 Business Models

Games can also be categorized based on the business model the development team employs. Game developers utilize various methods for generating revenue; that can be best applied to different platforms.

Direct Payment Models: While becoming less popular, retail sales are the traditional method for distribution of PC and console games (Adams, 2014). Games distributed in this manner are typically big-budget games with large publishers (Chandler, 2013). This method is not suitable for mobile game distribution. Digital distribution is similar to traditional; however, purchases happen through digital distribution platforms like Steam, App Store, and Xbox Game Store (Adams, 2014). Subscriptions require players to pay a monthly fee to access the game (Adams, 2014; Chandler, 2013). World of Warcraft, developed by Blizzard Entertainment, stands as a prominent example of this model. However, it can be claimed that platforms that implement "game pass" systems (Xbox Game Pass, Apple Arcade, EA Play) also adhere to this model. Episodic delivery and crowdfunding are also models, mostly utilized by computer game developers.

Indirect Payment Models: The Freemium (combination of free and premium) model consists of giving the basic game for free while supplying premium content for sale; which can be in the form of in-app purchases (IAP) or downloadable content (DLC) (Adams, 2014). The free-to-play model is similar, where the game is free, but purchases or subscriptions are required to advance in the game. Developers can also

utilize advertisements implemented in games to generate value (Adams, 2014). This model is the most prominent way to generate revenue in hypercasual games. Another revenue stream for studios is commissioned games, which are games made for someone else, where the developer gets paid by the commissioner to make the game (Adams, 2014).

2.2.3 Game Development

During game development, many disciplines work together in interdisciplinary collaboration. Art, engineering, and psychology disciplines get synthesized together to create games (Callele et al., 2005).

2.2.3.1 Roles

Game development teams usually consist of producers, artists, engineers, designers, and QA testers; although, a single person can also fall under more than one role in smaller teams (Chandler, 2013). While *developer* can be used to refer to the programmer/engineer, it can also be used to refer to a person who develops games regardless of their role. In this study, the game developer refers to the latter. Table 2.3 shows different departments and the roles within them. Main departments in game development can be listed as design, art, programming/engineering, production, audio, and quality assurance. Developers with diverse roles, while listed as departments in this table, typically collaborate as a team rather than dividing into separate departments.

Developers with design roles are responsible for the design of game mechanics, story of the game, and the overall gameplay experience (Chandler, 2013). Writer and other narrative roles can be placed with designers; however, they can also have their own department in larger game companies. Artists in game development are responsible for creating the visual elements of the game. They work closely with the developers with design roles (Chandler, 2013).

Table 2.3 Roles in Game Development Teams (Adams, 2014; Bates, 2004; Chandler, 2013)

Design	Art	Programming/ Engineering	Production	Audio	Quality Assurance
Lead Designer	Art Director	Lead Programmer/ Engineer (Tech Lead)	Executive Producer	Audio Director	Lead QA Tester (Test Lead)
General Game Designer	Lead Artist (Art Lead)	Programmers	Producer or Project Manager	Sound Designer	Testers
Mechanics Designer	Concept Artist		Associate Producer	Composer	
Level Designer/World Builder	Animator/Animation Artist		External Producer	Audio	
User Interface Designer	Technical Artist				
Writer	Marketing Artist				

Programming/engineering roles are responsible for technical aspects of the game development, including writing the game code, implementing mechanics, developing or integrating game engines, and optimizing the game performance (Aleem et al., 2016a; Aleem et al., 2016b).

Production roles are responsible for overseeing the entire development process. They monitor the progress of the project and act as intermediaries between the development team and external parties, such as publishers (Chandler, 2013).

Audio developers are responsible for creating and managing the sounds of the game. These sounds can include sound effects, voice-overs, and the music (Chandler, 2013).

The Quality Assurance department is responsible for testing the game to ensure its quality (Chandler, 2013).

For a successful game development process, these diverse roles need to work interdependently and efficiently (Aleem et al., 2016b; Caravella et al., 2023). The diversity of the roles indicates the complexity of the game development process (Aleem et al., 2016b). The distribution of roles within a game development team can differ according to the scope of the game project.

2.2.3.2 Publishers

Most game developers work with publishers to market and distribute their games (Chandler, 2013). For larger games on consoles and computers, studios can work with publishers on a project basis; in the hypercasual model, studios get contracts with publishers where they deliver a set number of games within the given time frame. Publishers take these games and advertise them to test their appeal. If the game succeeds, the proceeds are divided between the publisher and studio based on their deal. While most of these deals are kept private between the development teams and publishers, one example of a deal to be won in a challenge by CrazyLabs is posted online to attract developers (*55 per Cent Revenue Share in the New Hypercasual Developer Challenge by CrazyLabs*, 2022).

2.2.3.3 Game Development Pipeline

The stages of game development have been explored through different terminology; yet, they refer to similar concepts which can collectively be called the game development pipeline. Figure 2.4 shows a comparison between how researchers divide the game development pipeline based on the stages identified by Adams (2014), Chandler (2013), Bates (2004), and Greene and Palmer (2017).

Concept Stage	Elaboration Stage		Tuning Stage		(Adams, 2014)
Preproduction		Production	Testing	Post-production	(Chandler, 2013)
Concept Development	Preproduction (Proof of Concept)	Development			(Bates, 2004)
Concept Proposal	Concept Document	Technical Design Document & Game Design Document		Collected Feedback Document	Game Development Documentation Stages (Greene & Palmer, 2017)

Figure 2.4. Stages of Game Development synthesized by the author

The first stage, usually called the concept stage or preproduction, starts with the creation of the idea of the game. In this stage, in addition to generating a concept, planning and defining the game also occurs (Adams, 2014; Chandler, 2013). Chandler (2013) suggests the preproduction stage should take 10-25% of the development time. A game proposal, also called "pitch dock", is created at this stage to present to publishers (Bates, 2004). A similar document, the concept proposal, is also utilized to present the game idea to the team for approval to commence development (Greene & Palmer, 2016). With the approval of the game concept and the creation of the development plans, development proceeds to production.

In the second stage, production or development, asset and code production commence (Chandler, 2013). In this stage, the plan created during preproduction is implemented. This is the main stage where the game becomes playable.

The following stage is testing; however, some studies do not consider it as a separate stage but place it under production or postproduction. Chandler (2013) mentions the testing stage is critical; everything gets checked to ensure they are implemented correctly according to the plan. In the hypercasual model, this stage also includes testing for metrics such as CPI (cost per install) or player retention; a crucial step in determining the success of the game.

The final stage of the production process, known as postproduction or wrap-up, serves two primary purposes: gaining valuable experience and preserving the final product for future reference and distribution (Chandler, 2013). In this stage of development, a document called *post-mortem* may be created. This document summarizes "what went well" and "what went wrong" during the development process (Politowski et al., 2020).

The stages of game development can be understood as a cycle, and some steps may occur in multiple cycles in a single project (Chandler, 2013).

2.2.3.4 Project Management in Game Development

During game development, numerous tasks are contingent upon the completion of preceding tasks; therefore, this dependency must be taken into consideration while developing the plan (Chandler, 2013). Different software project management methodologies may be employed in game development projects, such as Agile and Scrum (Chandler, 2013; Greene & Palmer, 2016; Koutonen & Leppänen, 2013; McKenzie et al., 2019).

The Agile Manifesto was published in 2001 by 17 software developers who later founded the Agile Alliance. They describe Agile as a project management and software development strategy focused on flexibility, collaboration, and customer-centric iterations. It houses practices and frameworks, like Scrum, Kanban, sprints, etc. (*Principles behind the Agile Manifesto*, 2001). Agile development for games starts with creating a minimum viable product (MVP) version of the game and building the game throughout iterations (Chandler, 2013).

The Scrum approach was initially introduced by Takeuchi and Nonaka in 1986. They drew an analogy between their methodology and rugby, emphasizing the iteration; a multidisciplinary team engaged in iterative phases without adhering to a rigidly structured stage plan (Takeuchi & Nonaka, 1986). Following their work, two of the developers who published the Agile Manifesto, Sutherland and Schwaber,

subsequently developed the Scrum framework. They published the Scrum Guide in 2010. This framework focuses on production in short cycles called sprints, through iterative progress and team collaboration (Schwaber & Sutherland, 2010).

Three roles make up the Scrum team with no hierarchy: a Scrum Master, a Product Owner, and developers (artists, designers, and engineers) (Chandler, 2013). The Scrum Master works to make the project develop smoothly, while the Product Owner focuses on the design of the product and the creation of the tasks (Chandler, 2013; Schwaber & Sutherland, 2010).

A survey conducted among New Zealand game studies revealed that Scrum emerged as the most prevalent agile methodology, followed by Kanban and FDD (McKenzie et al., 2019). Their findings showed that studios do not adhere to agile methods fully, and they are often not aware of this. McKenzie et al. (2019) contend that the challenges that are associated with agile methods for not being suitable for a work may be attributable to the improper application of those methods. They also conducted further studies to investigate how agile methods are applied to game development. They interviewed video game studios to understand how and why they apply agile practices. The primary challenges game studios encounter within agile practices stem from the incorrect application of agile methodologies, compounded by team communication difficulties (McKenzie et al., 2021). Figure 2.5 shows how the agile methods were applied to the game development pipeline according to the results of their study; their research has revealed that game development operates on a dual-track methodology for the creation of mechanics and assets.

In addition to the Scrum methodology applied to game development, their study also reveals that the Kanban methodology is utilized in art development by some studios (McKenzie et al., 2021). Another study done with Finnish game studios also supports this claim by demonstrating that they use Kanban practices for game development on a small scale (Koutonen & Leppänen, 2013).

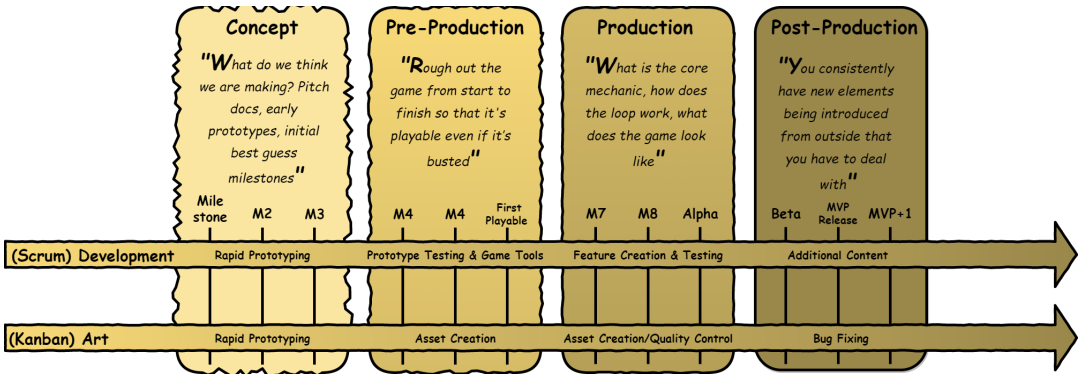


Figure 2.5. How Agile practices are applied to different stages of development (McKenzie et al., 2021)

Engström et al. (2018) argue that, as game development is a creative process distinct from software development, methodologies developed for software development may not be an optimal fit for game development. Chandler (2013) brings attention to a similar point; using Scrum or similar formal software engineering processes may not be as well-suited for design and art tasks. To investigate this mismatch, Hodgson and Briand (2013) studied the implementation of agile methodologies in a game studio. Interviews and observations with a Canadian game studio were utilized to examine the empowering claim of agile methodologies on creative teams. Their findings show that developers with creative roles evaded control technologies used to facilitate project management to employ personalized control, traditionally used in craft industries. The artists and animators participating in their study asserted that quantifiable methods are unsuitable for tracking the progress of art (Hodgson & Briand, 2013).

2.2.4 Hypercasual Game Development

Hypercasual games are typically built for mobile devices such as smartphones or tablets. They typically have a smaller scope than traditional video games, as they are designed with the device's limited resources in mind. This allows hypercasual games to be developed with smaller teams within shorter periods. On the other hand, hypercasual games are not a game genre but a business model in the game industry. They aim to create small and attractive games that cost less to make with high profits (*The Hypercasual Business Model*, 2021).

Small teams and short development times make hypercasual games fitting to study communication and collaboration between diverse roles. Hypercasual development involving rapid idea generation and iterations, causing team members to communicate and collaborate effectively, makes the process accessible to research. Combined with the interdisciplinary nature of game development, an opportunity to investigate communication and collaboration between diverse roles arises.

In addition to the development properties of hypercasual games, the gaming industry in Turkey also presents an opportunity for research. This industry has seen a significant growth in Turkey in recent years. With the emergence of high-valued companies and increasing market share, it can be said that Turkey holds a significant position in the global market and hypercasual gaming companies within Turkish digital industry also contribute to this growth (Le, 2022).

Hypercasual games are generally played by tapping or swiping on the touch screen and include basic gameplay. By utilizing simple gameplay, they aim to appeal to a diverse range of players. Their monetization model is generating revenue through in-app advertising, which requires a large number of downloads to generate value (Knezovic, 2024).

Figure 2.6 shows Fidget Trading 3D, a hypercasual game developed by MagicLab Game Technologies, which is a highly popular and successful game (*MagicLab Game Technologies*, 2024). The game utilizes tap and swipe controls and has a very

bright look and simple visual elements. The game shows a banner at the bottom of the screen and plays a skippable ad between levels, in addition to prompting players to watch more ads for benefits.

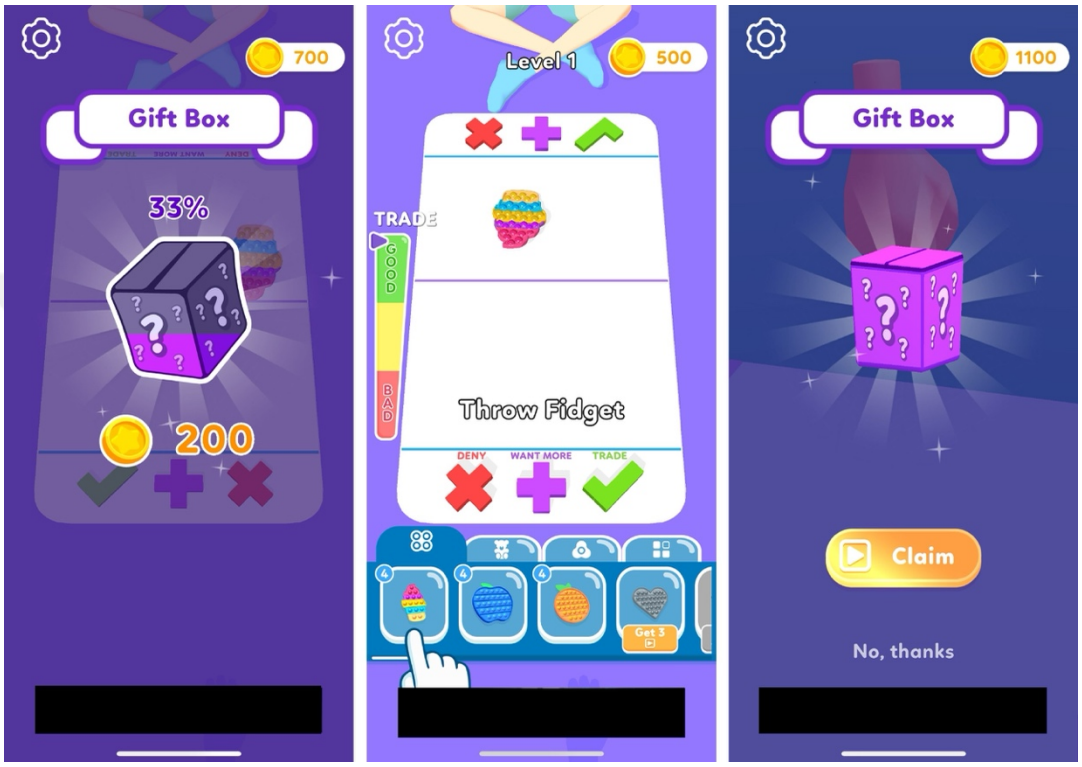


Figure 2.6. Screenshots taken from the game “Fidget Trading” developed by MagicLab

Hybrid-casual games share similarities with hypercasual games in their fundamental gameplay mechanics. However, they differentiate themselves by incorporating a more intricate gameplay experience, providing players with progression opportunities. They are designed to have higher user engagement and retention rates by utilizing “meta-game” elements, which may be in-game progression, social features, narratives, and competitive features (Knezovic, 2024; Pizzo, 2023). This

study places hybrid-casual games as a type of hypercasual games, based on the similarities.

2.3 Methods and Tools in Game Development

To better facilitate a study in collaboration while developing games, common practices in the industry must be discussed. Although the game industry is relatively young, the majority of commonly employed design and development processes are well-documented. In this part, the literature related to game development methods and tools is provided.

Almeida and de Silva's work (2013) categorizes different methods and tools used in game design under two main topics: efforts toward a shared design vocabulary and efforts toward a design visual language.

Efforts toward a *shared design vocabulary* consist of approaches like vocabularies and design patterns. The first suggested instance of a need for a shared vocabulary was "Formal Abstract Design Tools" (FADT), proposed by Church (1999). As the name suggests, Church argues design concepts should be identified in a formal and abstract way. An example given was "perceivable consequence" (Church, 1999, para. 30); "PERCEIVABLE CONSEQUENCE: A clear reaction from the game world to the action of the player.".

By identifying actions taken by the player and giving them common names, it is argued that games can be described and analyzed in a formal and abstract way, by using a common vocabulary.

Game design patterns are the generic name given to the recurring concepts that are reusable solutions to common problems or challenges in game development (Almeida & da Silva, 2013). Design patterns, which originated in architecture, were first suggested to be utilized in game development by Kreimeier (2003), drawing inspiration from Software Design Patterns by Gamma (1994, as cited in, Almeida & da Silva, 2013; Kreimeier, 2003). Bjork and Holopainen (2005) also brought an

approach to document recurring concepts of game design, called “Game Design Patterns (GDP) (Bjork & Holopainen, 2005). The project aimed to be a tool for game development and analysis. A newer version of the GDP, titled gdp3, is available online, consisting of 623 game design patterns (Patterns - Gdp3, n.d.). Although the GDP project has continued over the years, the limitations Almeida and da Silva (2013) mentioned persist; the wiki structure gives the source a high learning curve.

Within the “Games Ontology Project”, Zagal et al. (2005) propose a dictionary to create a common vocabulary. Their methodology seeks to discern the structural components of games and define the connections between them. Almeida and da Silva (2013) put vocabularies at a higher level of abstraction compared to patterns, which serve as a foundation for other approaches.

Within efforts towards a design visual language, examples like visual artifacts, canvas-oriented formalization, and one-page designs are discussed.

Canvas-oriented formalization of the game development process is a method that suggests the documentation of game design with the use of canvases for each aspect of the game (Luhova et al., 2019). These canvases are conceptualized as an information model that encapsulates creative decisions, thereby facilitating the generation of innovative design solutions. Series of canvases are created for areas like studies of consumers as players, studies of the aesthetics of the game based on the needs of the player, game mechanics and game dynamics, and research and design of the game world (Luhova et al., 2019).

One example listed under the search for design visual languages was presented in GDC (Game Developers Conference) 2010, One Page Design. Flowcharts, storyboards, timing diagrams, matrices, and tables were also given examples of One Page Designs. As seen in Figure 2.7, this format was suggested as a living document. The developers would use it to take notes, make changes, and share ideas with their team (*One Page Design - GDC Presentations*, 2010).

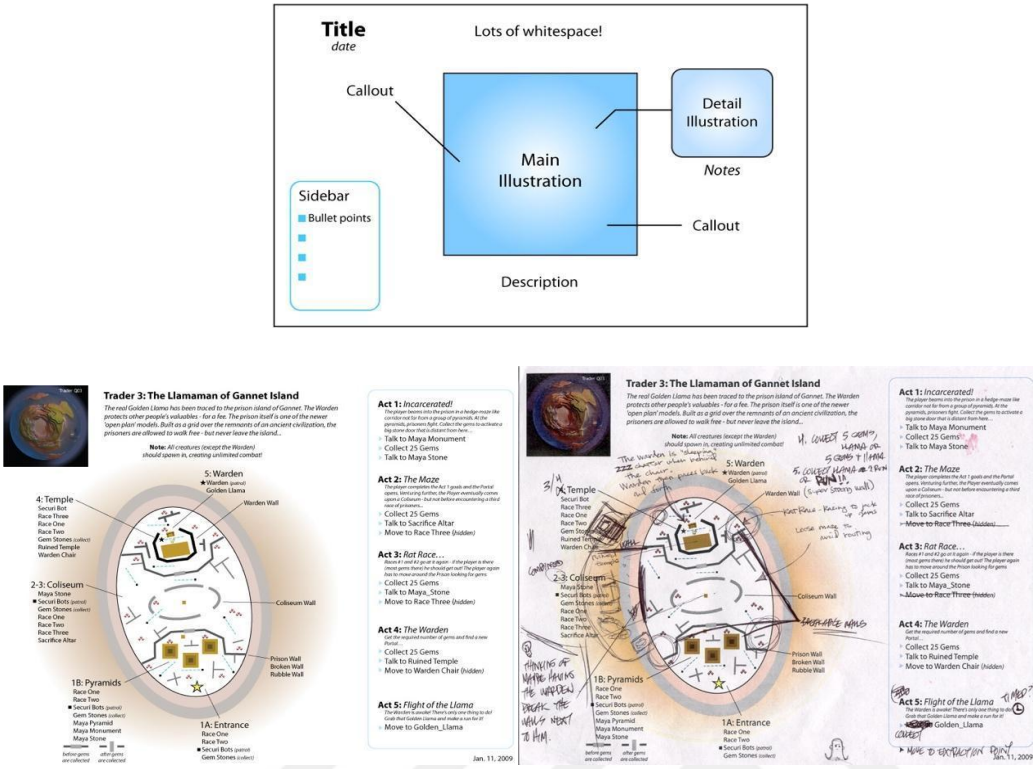


Figure 2.7. One Page Design: basic format, one page design when it was printed, and one page design after the designer used it during development. (*One Page Design - GDC Presentations*, 2010)

In addition to methods suggested in the literature for creating a shared vocabulary and design visual language, documentation is also used to communicate design decisions and project management tasks in game development. Table 2.4 shows documents encountered in game development documentation. Two main documents that are present during game development are Technical Design Documents (TDD) and Game Design Documents (GDD) (Chandler, 2013; Greene & Palmer, 2016). Bates (2004) also includes the Art Bible as a document to be created along with TDD and GDD.

Technical documentation is created by the lead engineer to document coding standards, technical design, and tool instructions; that serves as a guide for the programming team (Bates, 2004; Chandler, 2013).

Table 2.4 Documentation of Game Design During Development

High Concept Document	Concept Proposal & Concept Document	The Game Design Document
Game Treatment Document	Technical Design Document	The Art Bible
Character Design Document	Game Design Document	The Technical Design Document
World Design Document	Collected Feedback Document	The Project Plan
Flowboard		
Story and Level Progression Document		
On-screen Text and Audio Dialog Script		
The Game Script		
(Adams, 2014)	(Greene & Palmer, 2017)	(Bates, 2004)

GDD is a creative document prepared by designers in the team, it is crucial to game development as it documents various aspects of the game, such as the name, concept, genre, story, details of characters, gameplay mechanics, and levels, among other elements and details of the game (Callele et al., 2005). A GDD is presumed to be the outcome of the pre-production stage; however, it is expected to be maintained and kept up to date throughout the development (Aleem et al., 2016a; Bates, 2004). A well-managed GDD is found to be paramount for the development to move smoothly (Aleem et al., 2016b).

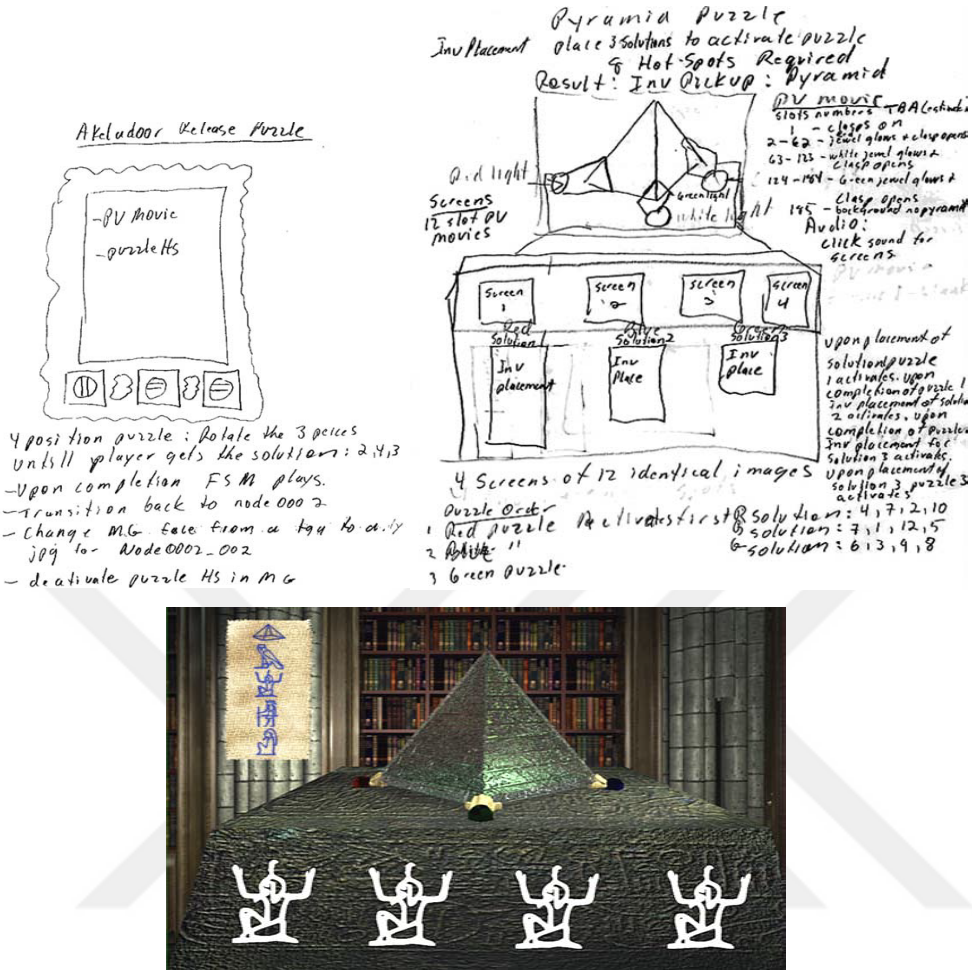


Figure 2.8. Three versions of puzzle drafts for the creation of Pyramid Puzzle (Callele et al., 2005, pp. 7-8)

In the example that can be seen in Figure 2.8, which is originally presented in Callele et al.'s (2005) work, the progression of documentation of one puzzle is seen. The first hand-drawn sketch is discarded while creating the next iteration, which shows how changeable the documentation can be while making design decisions. Between the iterations, the puzzle designer is presented with restrictions. The third image shows the prototype version of the puzzle. This example illustrates the necessity of cross-departmental feedback to effectively communicate programming requirements and enhance the development process (Callele et al., 2005). A GDD is expected to

be a living document, as exemplified by this progression of the puzzle. Feedback on the GDD and similar documents is collected on the files themselves throughout the development (Greene & Palmer, 2016).

While a GDD is a central document in development, different approaches are taken to improve its effectiveness.

Adams (2014) describes a different approach for GDD that utilizes multiple documents. The types of GDD in their approach can be listed as: high concept document, game treatment document, character design document, world design document, user interface design document, flow board, story and level progression document, on-screen text and audio dialog script, and the game script (Adams, 2014). These categories can be collected under one GDD or multiple documents based on the game's size.

Salazar et al. (2012) analyze different GDD versions and apply software requirements specification standards to create an improved GDD. Their contributions enhance the GDD approach by fostering a shared understanding, incorporating quality assurance measures, emphasizing decision-making based on tangible impacts, and clarifying relationships, boundaries, limitations, and the intricacies of game components (Salazar et al., 2012).

An alternative game documentation approach that takes a bigger role in development by automatizing some processes is also suggested. One example of such an approach is forming an <e-Game> language with a game storyboard. This is suggested as a system in which all the decisions made about the game are documented, as they are decided. Within this system, creating and keeping the storyboard is the main goal of the development process. With the language they suggest, the game will be automatically developed by processing the storyboard (Moreno-Ger et al., 2007). However, their system required a specific knowledge of development language, which was not suitable for every developer with different roles. Also, it only allowed one specific kind of game to be developed. Nevertheless, the documentation approach can be applied to game development without using the automatic system.

Documenting decisions while they are decided on the Game Design Document allows developers to leave the meetings with clear tasks.

While building a game does not necessitate a particular tool, asset type, or programming language, there are several software programs that are used more commonly in the game development industry. When talking about such software, Adams (2014) uses the term middleware and defines it as software that is pre-packaged and ready to use, or that requires minimal modifications. In the context of game development, this encompasses a comprehensive framework that manages various aspects of game development, such as sound, rendering, and physics simulation (Adams, 2014; Chandler, 2013).

Game engines are a type of middleware utilized in game development, serving as a replacement for the traditional approach of writing the game's code exclusively using a programming language (Christopoulou & Xinogalos, 2017). These types of software emerged in the 1990s to enable reusability of code libraries and assets in game development process (Mazzuca et al., 2022). They present developers with a wide range of tools and features, which allows developers to focus on the design of the game (Aleem et al., 2016b).

A comprehensive analysis conducted on the currently available game engines reveals Unity and Unreal Engine as the most potent engines, possessing nearly all the features categorized under the study's framework (Christopoulou & Xinogalos, 2017). According to the GDC's industry survey, Unity and Unreal Engine are the most widely utilized engines, each accounting for 33% of the market share, with Unity maintaining a slight lead (Game Developers Conference, 2024). Comparing the two, Unity is more suitable for beginners with its simpler UI, easy access to tutorials, and assets available within their asset store (Christopoulou & Xinogalos, 2017). These aspects also make Unity the best choice for developing hypercasual games.

Using middleware streamlines game production while also standardizing assets used by different projects using the same middleware. This allows some developers to

share with or sell game assets to other developers. Unity, for example, has the Unity Asset Store, which is a storefront that comes within the Unity engine. Game assets, including 3D and 2D models, audio, visual effects, and add-ons that provide extra functionality to the engine, are available for purchase or download from the website. Most hypercasual games utilize assets from this store, as it shortens the development time.

2.4 Collaboration in Game Development: Challenges and Suggestions

The challenges that game developers encounter during the development process stemming from communication and collaboration can be categorized based on team dynamics, communication, tools, and processes, which demonstrates the need for enhancing collaboration and related tools.

Team dynamics can be investigated under issues based on multidisciplinarity, team configuration and management, and insufficient workforce. Game development is an interdisciplinary and multidisciplinary process where professionals from various backgrounds collaborate, and they need to manage this process effectively. To ensure a smooth collaboration, Ahmed et al. (2017) suggest that all team members should get training on how to work together as a team, based on divergent viewpoints which developers acquire from diverse professional backgrounds and varied interests. It is also suggested that team members need to have a shared understanding of game vision during game development to facilitate coordination and cooperation. Liu and Liu (2020) investigate how shared understanding influences project success, based on the case study of a Chinese game development company. Their findings indicate that a well-established concept, shared by all essential team members, is crucial for effective coordination.

Team configuration and management is another challenge that is related to the multidisciplinary nature of game development. Studies conducted by Aleem et al. (2016a, 2016b) and Politowski et al. (2021) bring attention to the importance of team

configuration for effective collaboration. To introduce a novel digital game maturity model, Aleem et al. (2016a) examine the game development process and define five levels of maturity. The initial level of maturity is designated as the "Ad-Hoc" level, which signifies that the game development organization lacks a structured and consistent methodology. Through opportunistic, consistent, and organized/predictable levels, the development teams start to employ best practices and well-defined guidelines. The highest level is optimized, and game development organizations in this level development process are managed and conducted effectively. The model examines common management and development practices to assess game development teams. At lower maturity levels, they observed the absence of a well-defined team configuration and effective management (Aleem et al., 2016a). Another study by Aleem et al. (2016b) presents a quantitative study to investigate critical success factors in the game development process from the developers' perspective; that identifies team configuration as a key factor of achievement. Politowski et al. (2021) adds to their findings by suggesting creating mixed teams with various disciplines instead of teams based on disciplines to facilitate the effective development process. Their study analyzed "post-mortem" reports from game development projects between 1997 and 2019 to identify and categorize recurring development problems. Based on their findings, one suggestion they have proposed is to reduce team sizes while ensuring that each team comprises all the necessary disciplines. This approach aims to create teams that possess the capability to develop all aspects of the game within themselves (Politowski et al., 2021).

Another challenge emerging in team dynamics is stated to be caused by insufficient workforce, based on the work of Politowski et al. (2021). Their findings suggest that issues caused by poor project planning can lead to "feature cutting" and "crunch time". Feature cutting is removing previously planned features during the development process due to time constraints or budget limitations (Ullmann et al., 2022). Crunch is another problem that is widespread in the game development industry; it refers to extra hours spent on working on the project (Lehtonen et al.,

2020; Politowski et al., 2020). They can be both solved by better management of complex tasks, rather than hiring more developers (Politowski et al., 2021).

Communication issues in game development may be caused by three factors: differences in communication styles, coordination difficulties, and work environment.

Differences in communication styles may result in communication issues in game development. These differences can be caused by the interdisciplinary nature of game development, different technical backgrounds, or a lack of formal communication protocols. The first two causes are also discussed under the team dynamics category. A team composed of diverse disciplines may possess varying communication styles, vocabularies, and work methodologies, which can potentially lead to the erosion of an effective development process (Lehtonen et al., 2020). Lehtonen et al. (2020) investigate the adoption of requirements engineering methods in game development by examining game developer post-mortems shared online. They identified a general lack of processes and methods, or the unsuitability of methods imported from other software development fields as issues of game development.

Coordination difficulties may also lead to communication problems in game development, as it is also presented in Liu and Liu's work (2020).

Communication issues caused by the work environment refer to ambiguity regarding work boundaries and responsibilities, which can result in conflicts. McDaniel's study (2015) features examples related to work boundary issues. For instance, the relaxed environment of a game studio might lead new employees to overstep boundaries if the organizational culture isn't clearly communicated to them. Their study delves into a case study involving interviews with developers over a three-month period. It explores communication and knowledge management strategies within a game development studio. It has been identified that there is an area for improvement in the area of communication regarding work boundaries (McDaniel, 2015).

Tools and processes used in game development may also result in communication problems. Use of tools that are not specialized for game development may hinder communication among team members. Hossain and Zaman (2023) list the types of tools used for project management, version control, instant messaging, and video conferencing. Their work focuses on the need for a tool that supports synchronous and asynchronous collaboration, version control, and instant messaging along with NCCollab which is a software they developed to address these requirements. NCCollab is a node-based behavior authoring framework that is an extension of NodeCanvas for the Unity game engine, which facilitates collaborative visual programming of game AI between multiple developers in synchronous and asynchronous modes (Hossain & Zaman, 2023).

Tool selection is also important in the sense that game development tools can directly influence the design and development processes. One issue presented by Politowski et al. (2021) is that game engines can be a source of frustration in game development, as they can constrain game design. Inappropriately chosen game engines may also cause delays and rework in the development process (Politowski, 2021). Frustration caused by reworks could affect the collaboration during the development process.

The utilization of version control systems can also lead to communication difficulties. Although their adoption facilitates collaboration in software development, issues related to awareness stemming from opaque practices may also arise. It is stated that lack of transparency in tracking changes can hinder collaboration (Chandler, 2013).

2.4.1 Game Development After the Pandemic

Remote work has been a topic of interest across various disciplines, and a significant increase in interest can be observed due to the COVID-19 pandemic (Huang et al., 2024; Yang et al., 2022). Remote collaboration is explored through the lens of communication and the technologies that facilitate it.

Remote collaborations bring additional challenges to collaboration; such challenges may arise from the physical separation of collaborators and the reliance on technology to mediate interactions (Tanner & Naidoo, 2021). Tanner and Naidoo's study (2021) investigates tools utilized by agile software development teams to enhance communication and collaboration during work-from-home arrangements. Through in-depth case studies and semi-structured interviews, they investigate how specific tool features were perceived and used. Their research resulted in four outcomes that emerged from the realization of different affordances: fostering engagement, streamlining communication, enhancing virtual meetings, and increasing shared understanding (Tanner & Naidoo, 2021).

One notable challenge remote working may pose to collaboration is the potential loss of shared ground. Collaborators who are physically present in the same location may be able to establish a shared understanding through verbal and nonverbal cues, which is diminished in remote settings (Huang et al., 2024). A similar key challenge in remote collaboration can be limitations of communication channels (Oliveira et al., 2015). Computer-mediated communication is considered to be less rich compared to face-to-face communication (Huang et al., 2024; Oliveira et al., 2015). The task space being separated from the communication space could lead to a loss of work awareness. Text-only and audio-only communications can slow the communication due to a loss of communication elements like nonverbal cues (Huang et al., 2024).

The issues related to remote work may also be an important consideration for game development. The game development industry had been utilizing remote work even prior to the pandemic, but the practice became more prevalent after the pandemic (Game Developers Conference, 2024). It is also important to note that remote work prior to the COVID-19 pandemic is different from the mandatory work from home as workers also have other responsibilities they may need to attend to at home (O Connor et al., 2022).

Challenges stemming from remote collaboration include social isolation and reduced team cohesion (Timmermann et al., 2022). One particular study by Caravella et al.

(2023) looks at how remote work affected communication and collaboration practices by surveying International Game Developers Association (IDGA) members on the impact of COVID-19 and remote work. Their findings state that the sense of productivity of developers is not affected; however, they report a substantial decline in mental well-being due to social isolation and exacerbation of the responsibilities of household caregivers (Caravella et al., 2023).

CHAPTER 3

METHODOLOGY

This thesis aims to examine the ways to enhance design communication and collaboration between team members during mobile game development. To fulfil this aim, a research study comprising two stages has been conducted. This chapter presents the methods employed to study communication and collaboration between team members during game development. Interviews are conducted to gain a better understanding of the needs of game development teams, and generative workshops are done to better examine the discovered issues. By utilizing interviews and a generative workshop, qualitative data relating to the game development process and communication and collaboration within this process were gathered.

3.1 Research Approach

This study employs qualitative research methodologies, including interviews and a workshop, to gather data on game design and development process, and collaboration tools and methods employed by the game developers. Saldana (2011) defines qualitative research as an overarching term that encompasses collecting and analyzing data about human experiences. Additionally, they put meaning in context at the focus and employ sensitive methods that garner underlying meaning while collecting data (Saldana, 2011). Creswell (2009) further explains qualitative research as a process to extract the meaning of social or human problems by collecting data through questions and analyzing responses by making interpretations of the meaning of the data.

This study is designed to answer these three research questions:

Q1. What content is being communicated between the members of the game design team throughout the (game) development process?

Q2. What are the strengths and limitations of the existing tools and/or methods utilized in mobile game development for design communication and team collaboration?

Q3. How can the communication of mobile game design and the collaboration between team members be supported by design during the game development processes?

By applying qualitative research methodologies, it is possible to attain an understanding of individual and social complexity (Saldana, 2011). Communication and collaboration between teammates with the same and different roles during the game development process can be explored through qualitative research, as it is a means to understand the underlying meaning behind human behaviors (Creswell, 2009). Therefore, qualitative methods were chosen for this thesis research to understand the underlying reasons for behavior presented during game design and development.

A two-part study is conducted to address the research questions. To understand personal perspectives, semi-structured interviews were conducted with 11 game developers that represent different roles in game development teams. The interviews were followed by a workshop session, which was used to detect common communication and collaboration problems experienced by developers with different roles and their methods to overcome these problems.

3.2 Research Stages

To investigate the questions above, two main methods have been employed. In the first part of the study, semi-structured interviews were used to learn about the behavior patterns and routines of developers and the tools and methods they employ. The second part of the study utilized a focus group and generative workshop

approach to discuss experiences collectively to find design intervention areas and generate ideas appropriate for the challenges in communication and collaboration during game design and development.

3.2.1 Sampling

Selection of people to represent the population of interest is referred to as sampling (Sanders & Stappers, 2012). Purposeful sampling methods, a type of non-probability sampling, are used in qualitative research as they involve choosing participants that can provide insights on relevant experiences to the research question (Koerber & McMichael, 2008). This makes purposeful sampling suitable for this study. The first step in purposeful sampling is choosing the selection criteria (Merriam, 2014). As communication between team members with different roles is to be studied, it is necessary to include professionals with different roles in the game development team. The inclusion criteria for the study are as follows:

Game developers;

- Working with people with diverse roles in their team;
- With at least one year of experience working in a game development team;
- Developing hypercasual or casual mobile games.

Participants were invited to the study in several different channels, including social media (Figure 3.1) and our department's mailing list, which included brief information about the research and inclusion criteria. In addition to these, printed invite flyers (Figure 3.1) were distributed in ATOM (Animation Technologies and Game Development Center) in METU Teknokent, which is an incubation center for game developers. Individuals who have expressed interest in participating are subsequently evaluated to ascertain their eligibility for the criteria. As most of the participants were located in diverse locations, they joined the study via online methods, and the consent form was sent through email (Appendix A). In addition to the purposeful sampling which enabled reaching participants via these channels,

snowball sampling was also utilized. Participants who responded positively to the research invitation and attended the interview session were asked to reach their co-workers or their friends who fit the criteria.



Figure 3.1. Social Media invite & Printed Invite

Game developers from different disciplines (art, design, programming, marketing) at different levels (junior, mid, lead) were included in the study. All participants were interviewed; 5 of the participants had participated in the workshop. Table 3.1 presents the role distribution, demographics, and participation in greater detail.

Before starting the study, ethics approval from METU UEAM (Protocol Number: 0378-ODTUİAEK-2023) was acquired. Participants were sent informed consent forms via email for each part of the study separately. They were also asked for verbal consent before each recording.

Table 3.1 Participant Information

	Current Job Title	*Work System*	*Time with Current Team*	*Gender*	*Age Group*	*Interview Type*	*Workshop*
P1	Game Artist	On-site	2 years	Female	26-35	On-site	Yes
P2	Product Specialist & Game Designer	On-site	1,5 years	Female	26-35	On-site	Yes
P3	QA Team Lead	Remote	2 years	Female	26-35	Online	Yes
P4	Game Designer	Hybrid	1 year	Male	18-25	Online	
P5	Marketing Artist	Hybrid	4 months*	Male	26-35	Online	
P6	Gameplay Designer	On-site	2 years	Male	26-35	Online	Yes
P7	Co-Founder, Game Artist	On-site	2,5 years	Male	26-35	On-site	
P8	Software Developer	Remote	4 months*	Male	26-35	Online	Yes
P9	UX Designer	Hybrid	1,5 years	Female	18-25	Online	
P10	Game Developer (Junior)	On-site	2 years	Male	18-25	Online	
P11	Game Developer (Mid-level)	Remote	2,5 years	Male	26-35	Online	

*: Participants with previous experiences of game development in their former teams (P5: +2 years; P8: +1 year)

3.2.2 Interview Design & Conduct

Semi-structured interviews, which are interviews with a guide that also includes flexibility in their design, were utilized to gather in-depth information from team members. This type of approach was suited to the study as it allows capturing participants' unique experiences and perspectives; as the participants represent different roles in game development (Rabionet, 2011). The semi-structured

interviews included a combination of specific and open-ended questions, prompting additional questions to understand underlying reasons behind the participants' behaviors and experiences (Merriam, 2014). Semi-structured interviews were conducted with participants to gather their experiences and insights into the game development process. The interviews were concluded with a question list (Appendix B); additional questions were asked to further understand the participants' game development processes. The interviews started with providing participants with information about the study's topic and the interview, and they provided their informed consent (Appendix A) to attend the study. Questions in the interview were gathered under two main categories with an additional category for demographics questions. The first category inquired about the game development process the participant and their workplace employ. The type of game developed, which method is employed during development; and tools and methods utilized were inquired in the first questions. The second category starts with inquiries on which roles are present in the team, how is the communication between team members with diverse roles is, and what kinds of meetings occur within the team. These questions were followed by the content shared within the team, examining types, channels, feedback, and how the content relates to disciplines like art, design, coding; focusing deeper on appropriateness of the tools and methods employed for communication and collaboration. Interviews were concluded by requesting the participant to describe their ideal communication and collaboration tool for implementation in game design and development.

The interviews were conducted either online or face-to-face based on participants' preference and availability. Out of 11 interviews, 8 participants joined via online methods, and 3 participants partook face-to-face. A pilot session was conducted beforehand and was also included in the data set to be analyzed, as the participant provided valuable data that contributed to the study's objectives.

Both online and face-to-face interviews were recorded with the participant's consent. Online meetings employed communication apps Zoom and Google Meet. Both applications allow the meeting lead to record audio and video. The interviewer's and

the participant's video were on in most interviews, but only the audio recording was kept and analyzed. Face-to-face interviews were recorded using the Voice Recorder application on the smartphone and the computer to create a backup recording. Sessions ranged in duration from 30 to 75 minutes, with the majority lasting approximately 50 minutes.

3.2.3 Analysis of Interviews

3.2.3.1 Transcription

The recordings of interviews were transcribed using MacWhisper software, which utilizes Whisper. Whisper is a product of OpenAI which does automatic speech recognition (*Introducing Whisper*, 2024). This approach was selected for several reasons. The first factor was that using the application made the process faster than transcribing manually. Nevertheless, corrections were also needed, which took 1.5 times the recording length. Secondly, the application transcribes recordings on the device rather than transferring them to another platform, which is important to keep participants' data safer. The third factor was that the application provides subtitle files (.vtt) which can be imported to MaxQDA, the qualitative coding software, with the audio recording. This allowed the researcher to jump to specific parts of the recording during data analysis.

3.2.3.2 Analysis

After the recordings were transcribed, data analysis started. In this study, thematic analysis methods are employed. Thematic analysis is conducted by identifying patterns of meaning (themes) within the qualitative data. Then, themes are examined to identify patterns (Clarke & Braun, 2017). This process is also described in chapter 4 of the book Design Methods (Matthews & Ross, 2010).

While analysing the participants' responses, first, the transcripts of interviews were read, and excerpts relating to research questions were highlighted to become immersed with the data. Thus, the initial codes and themes are generated, while also incorporating themes from the literature. Transcriptions and initial themes were imported to MaxQDA, which is a qualitative analysis software. Codes are sorted into potential themes relevant to research questions. Figure 3.2 shows the file organization and the initial codes in MaxQDA software. Then, themes are refined by reviewing coded excerpts using the software. The excerpts and themes from this step are exported to Miro. Finally, a hierarchical code tree of themes is generated. The Figure 3.2 shows the code tree and excerpts in Miro.

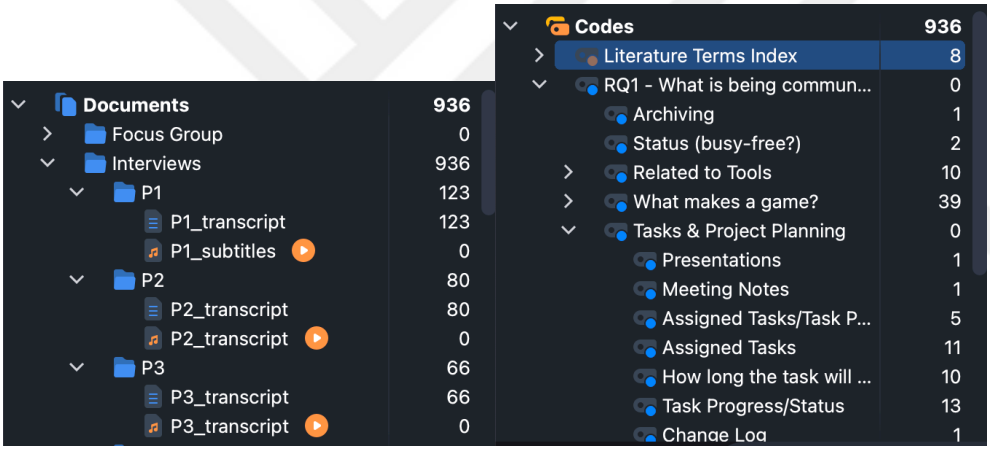


Figure 3.2. File organization and initial codes in MaxQDA Software

3.3 Generative Workshop

3.3.1 Generative Workshop Design and Conduct

The second part started after the initial themes of the first stage are identified. Findings are used as a guide to construct the generative workshop session. This

session was designed as a focus group integrating generative workshop methods, which will be referred to as workshops. In this session, participants were presented with scenarios similar to the ones they encounter daily, and they were prompted to react to these situations.

Generative workshop sessions were made to explore common problems the participants encountered during game development related to communication and collaboration and to discover possible intervention areas. By employing an additional methodology, *methods triangulation* was sought to achieve a comprehensive understanding of the subject matter and enrich the findings (Sanders & Stappers, 2012).

The workshop session contained generative elements; a toolbox was created which included images, drawings, and words related to the game development process; aimed to stimulate expression and gather user insights (Sanders & Stappers, 2012). During the iterative development of the toolbox, trigger items are systematically collected and reviewed to refine the toolbox for optimal effectiveness. Prior to the creation of the toolbox, instructions for the workshop were written. A list of words was compiled, which then were used as a guide to obtain visual materials. The word list and obtained visuals were reviewed together to detect if additional elements are needed. Additions were made, and ultimately the toolbox is finalized. To create an appropriate toolbox, steps described in Sanders and Stappers's work (2012) were utilized. Figure 3.3 shows the overview of the toolbox, and the detailed visuals can be found in the Appendix D.

As the participants resided in different cities, Google Meet was chosen to bring them together. The session commenced with a single meeting; however, three meetings were subsequently held to facilitate tasks that required collaboration in pairs during the study. Google Meet's recording feature was utilized; the session was recorded with the participants' consent. Consent forms were sent to participants prior to the study and were collected before starting. The workshop lasted three hours, which was the time frame given to the participants.

The session started with presenting the study to the participants. A quick overview of features of Miro, which will be used during the study, was presented. The session consisted of 4 steps: introduction, immersion, activation, and brainstorming.

The first step, introduction, was a step where the participants got familiar with the Miro interface and introduced themselves to the other participants. The second step aimed for the participants to reflect on the game development process they experienced at their workplaces. They were given a timeline frame, in which they would create a collage of the game development process, utilizing the toolbox. After creating the collages, participants presented their work and made comments on the process. In the next step, participants were paired, whose roles were taken into consideration to simulate communication between diverse roles. They were asked to create a comic strip presenting a commonly occurring problem that happens during communication and collaboration (Figure 3.4, Appendix F). They were provided with drawings of people, office elements, and speech bubbles. This step also was concluded with presenting to the group. This step aimed to activate the participants' memories of where pain points occur in communication and collaboration during the game development process. Additional materials provided in this step also included screen captures from widely used development tools, which the participants were requested to incorporate if necessary. In the last step, participants were asked to write similar problems they encountered on sticky notes (They were also encouraged to write how they cope with such problems or how they think these problems can be managed). After doing this individually, all of the sticky notes were copied to a bigger shared frame to find similar points together with discussion. With this, the participants were able to give more detail on problems they noted on the sticky notes.



Figure 3.3. Toolbox used in the generative workshop (Appendix E)

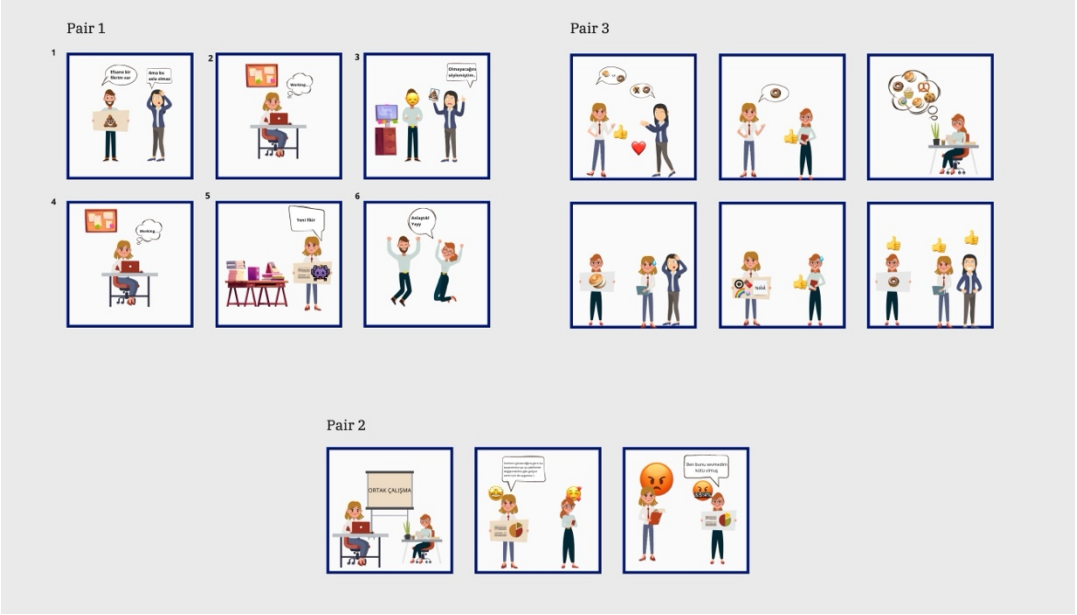


Figure 3.4. Comic strips created by the participants (Appendix F)



Figure 3.5. Groups made by participants (Appendix F)

3.3.2 Analysis of Generative Workshop

The analysis of the generative workshop started during the session. The researcher and the participants grouped the similar issues and suggestions together (Figure 3.5, Appendix F). After the analysis of stage one was fully completed, the outcomes of the workshop were revisited. Issues and suggestions were grouped into the themes that emerged from the previous stage.

3.4 Limitations

The principal limitation of this study is the number of participants. A study with a larger number of participants may have further endorsed the validity of presented

results; however, the results obtained can still be considered valuable in understanding this subject.

Another limitation was the result of role imbalance amongst the participants. Roles such as game artists or developers were relatively numerous, while fields such as QA or marketing are less represented. This distribution is somewhat representative of the hypercasual game development, but different insights can be obtained by also analyzing collaborative needs for other roles.

Finally, the logistical limitations of the participants resulted in some interviews being conducted online while others were conducted face to face. This is not dissimilar to actual working conditions of the participants; however, environments of online interviews are more difficult to control, which may increase the variation of participants' responses. The weakness of online interviews is most observable for the workshop study, where sharing a common physical space may have enhanced collaboration.

CHAPTER 4

FINDINGS & DISCUSSION

This chapter presents the findings of the two stages of the study. The first stage is the semi-structured interviews. The findings in this stage are grouped in terms of the aspects of development they are related to. These aspects are task management, communication, collaboration, and documentation; followed by concerns related to particular tools. The second stage conveys the findings of the generative workshop study. These findings are also grouped under aspects discussed in the first stage.

4.1 Stage 1 Findings

The first stage of this study utilizes semi-structured interviews to understand participants' game development process and collaboration with their teams.

The interviews also revealed the tools participants utilize during game development. The tools were examined and divided into two categories: individual use tools (Figure 4.1) and collaboration tools (Figure 4.2). Individual use tools are mostly used to create game assets and game elements like 3D models, UI elements, code, and audio files. Game data tools that offer information on game marketing were also filed under this category. According to McGrath's (1991) team behavior model, tools under this category facilitate production behavior. On the other hand, tools under the collaboration category are used for coordination purposes and to facilitate articulation work.

In the literature, software that facilitates and supports cooperation in teams is discussed under the CSCW research area. The categorization of technologies supporting collaborative behaviours presented by Grudin and Poltrock (2014), as discussed in 2.1.2., can also be used to cluster the tools that support these behaviors.

Figure 4.2 shows the tools mentioned by participants that facilitate collaboration, which was categorized within this framework. Time is a dimension that separates synchronous and asynchronous; however, modern tools with large scopes are capable of supporting both. In the example of whiteboard/infinite canvas tools like Mural and Miro, which can support synchronous collaboration, they can also be used as document repositories, holding information related to game design to be accessed on demand. The second dimension is content, and there also exist tools that offer more than one type of content. Tools like Google Workspace and Microsoft Teams can be regarded as toolboxes as they bring together many different functionalities, like meeting facilitation, calendars, video and audio calls, document repositories, meeting the needs of all three types of content.

Tool	Type	Tool	Type
Paint*	Meeting Support	Sketchbook	2D Art
Microsoft To Do	Task Management	After Effects	Video
Notes/Notepad	Task Management	Premiere Pro	Video
Obsidian*	Task Management	Mixamo	3D Art, Animation
3ds Max	3D Art	Spine 2D	2D Art, Animation
Blender	3D Art	Adobe XD	UI
Maya	3D Art	Figma	UI
Rhinoceros	3D Art	Sketch	UI
Zbrush	3D Art	Rider, Visual Studio, Xcode	IDE, Workflow
Adobe Animate	2D Art, Animation	Adjust, Admost, AppMagic	Marketing
Illustrator	2D Art	Facebook Ad Library	Marketing
InDesign	2D Art	Firebase*	Marketing
Photoshop	2D Art	SensorTower, StoreGlide	Marketing
Procreate	2D Art	Bitbar, Gamebench	Testing

Figure 4.1. Tools used individually by the participants

Tool	Type	Content	Time
Microsoft Teams	Audio, IM, Media Spaces, Text, Video + Audio, Workflow	Communication, Coordination, Information Sharing	Both
Google Workspace	File Hosting, Project Management, Text, Video + Audio, Workflow	Communication, Coordination, Information Sharing	Both
Trello	Project Management, Workflow	Coordination	Async
Monday	Project Management, Workflow	Coordination	Async
Jira	Project Management, Workflow	Coordination	Async
Google Calendar**	Group Calendars, Task Management, Workflow	Coordination	Async
ClickUp	Project Management, Workflow	Coordination	Async
Asana	Project Management, Workflow	Coordination	Async
Unity 3D	Game Engine	Information Sharing	Async
Word/Docs	Meeting Support, Repositories of Shared Knowledge	Information Sharing	Both
Powerpoint/Slides	Meeting Support, Repositories of Shared Knowledge	Information Sharing	Both
Mural	Digital Whiteboard, Meeting Support	Information Sharing	Both
Notion	Project Management, Repositories of Shared Knowledge, Wikis	Information Sharing	Both
Miro	Digital Whiteboard, Meeting Support	Information Sharing	Both
Google Drive	File Hosting	Information Sharing	Async
Git Tools	Version Control, Workflow	Information Sharing	Async
Excel/Sheets	Meeting Support, Repositories of Shared Knowledge	Information Sharing	Both
Confluence	Repositories of Shared Knowledge	Information Sharing	Async
Dropbox	File Hosting	Information Sharing	Async
Zoom	Audio, Video + Audio	Communication	Sync
Telegram	IM, Text	Communication	Sync
Slack	Audio, IM, Media Spaces, Text, Workflow	Communication	Sync
Google Meet	Audio, Video + Audio, Workflow	Communication	Sync
Discord	Audio, IM, Media Spaces, Text, Video + Audio	Communication	Sync

Figure 4.2. Tools used for collaboration

Another categorization made in CSCW literature groups studies conducted in the research area by Olson and Olson (2012); the grouping could be modified to differentiate features in tools used for cooperation and collaboration in game development, which can be seen in Figure 4.2. Tools mentioned by participants are

grouped related to their features that support collaboration. This demonstrates that tools can specialize in a specific category of features, such as Discord's focus on technical infrastructure; or they can provide features from multiple categories, as exemplified by Notion's integration of features from both coordination support and information repositories.

Participants' answers about their game development process and the tools they utilize revealed patterns which can be categorized under five titles: project and task management, documentation, communication, collaboration, and related to tools.

Table 4.1 Strategies and Dimensions of Project and Task Management

Project and Task Management	
Common Task Managers	Need for Common Task Managers in Game Development
	Factors Hindering the Use of Common Task Management
	Notifying Team Members About Tasks
Alternative Means of Task Tracking	Task Distribution via Communication Apps
	Tracking Development Progress Through the Game
	Calendars as Task Management Tools
	Tracking Changes Through Change Log
Task Differentiation	Through Task Weight
	By Visualizing Task Status
	Based on Development Stages
Personal and Team-level Management	

4.1.1 Project and Task Management

Project management is considered a part of game development, involving organization, planning, and communication (Maxim & Ridgway, 2007). This is also expressed by participants during interviews. P11 also highlights that they had taken a course on project management during their software engineering education.

One potential pitfall in project management is prioritizing the product's development over the project management itself. Project management serves as a means to facilitate and support articulation work, which involves coordinating efforts for collaborative work (Schmidt & Bannon, 1992). Project management is mentioned to be overlooked in order to finish the production on time by P8. In hyper-casual game development, the same production process is repeated almost every week, delivering as much as three games in one month to the developer. P8 highlights this by saying "*While staying organized and calm provides at least long-term advantages, unfortunately in Hyper-Casual, since we take a serious crush and compete with time anxiety, instant messy work is actually much more efficient.*"

Adaptation of common task managers, task tracking via different media, how tasks are differentiated, and personal and team-level management are examined within project and task management; which is summarized in Table 4.1.

4.1.1.1 Common Task Managers

Adapting a common task management tool for the team enables developers to monitor the progress of tasks, assign tasks, and maintain awareness within a structured framework. Improved communication and workflow are highlighted as key advantages, preventing duplicated effort and facilitating collaboration, especially in remote teams, through features like assignments, deadlines, and comments. However, the adoption of such managers is hindered by individual preferences for task management, inadequate implementation, and lack of team lead

mandates. There is a need for refined notification systems within task managers to balance efficient task assignment and minimization of disruptive interruptions.

Need for Common Task Managers in Game Development

Task managers allow developers to eliminate the overlapping of tasks, facilitate communication while remote working, and ensure team leads' control over the process. In the interviews, in addition to tracking one's own tasks, the most important features of task managers mentioned by P1, P2, P3, P8, P9, and P10 are assignments, deadlines, and comments. Assignments show which tasks are expected from whom, and in some cases, what task is being worked on. As tasks can be added and assigned at any time in the development process, when new tasks are added, assignees get notified. In addition, completed tasks also alert the person who is assigned to the next task, automating the process and creating a workflow. Moreover, showing who is working on the task prevents other designers from taking on the same tasks, eliminating overlaps. Features that show who is working on which tasks can be valuable during game development communication with this role of bridging tasks and preventing unnecessary work. While it was mentioned less in the interviews, setting deadlines on tasks is also added to this workflow. Deadlines can allow developers to organize their time more efficiently, since time is a valuable source during hypercasual development due to its high-paced production. Adding comments to task managers has a prominent role during game development communication. Especially in remote working, but not limited to it, comments help the developers directly ask their questions on the tasks. They lead to further discussions and creation of new design ideas by giving developers a focused space they can communicate.

Factors Hindering the Use of Common Task Management

In certain game development teams, the utilization of a common task manager is not implemented. This can be attributed to various factors, including individual preferences for task-tracking methods, inadequate implementation of the task manager, or simply not being mandated by the team leader.

The interviews reveal that everyone's approach to taking notes and following work tasks is unique, which may result in inconsistencies in task management.

Participant P6 compares their task notes with a colleague: "*I've seen people use the Notes app much more effectively on the computer, though. There was this thing with a checkmark feature or something like that. ... My friend was so organized that I envied them, you know, putting a tick or something, making a note under something else. They used it very well.*" P8 and P10 mention trying additional applications like "Taskmaister" and "Obsidian" to track their personal work. In P7's team, a common task manager is used to present developers with draft task lists, which then the developers are encouraged to make their own notes of tasks.

Participants P7, P8, and P10 find employing a common task manager intricate and cluttered. When a task manager is mandated inadequately, developers fail to utilize it; if developers are not checking if they are assigned tasks or marking tasks done, the task manager has no benefit to the game development process.

In the absence of a mandated task manager assigned by the team lead, developers tend to manage their own tasks using their own methods. In such situations, developers may explore alternative methods to manage their tasks. These methods include utilizing various applications, manually writing down tasks on paper, or attempting to recall tasks without any notes.

Notifying Team Members About Tasks

As previously mentioned, task managers can be used to notify other developers of task status. Using task managers can help developers get notified in their own way. While P1 emphasizes the importance of direct communication with the assignee in addition to task creation, P8 finds this approach distracting and feels rushed, therefore favors the notifications provided by the task managers.

The factors that impede the implementation of common task managers further highlight the significance of taking user expectations into account when designing a

task manager. A key concern is the need for improved task assignment and reminder systems, suggesting a shift from constant interruptions to more targeted notifications.

4.1.1.2 Alternative Means of Task Tracking

Task progress tracking is not limited to the use of task management applications. The interviews reveal several alternate methods: utilizing communication applications, tracking through the game, time-blocking in calendars, and creating internal change logs.

Task Distribution via Communication Apps

As P3 highlights, communication apps, primarily Slack, help with task distribution and progress updates through threaded conversations. Integrations of communication and task management tools allow developers to be notified about their tasks and ask questions directly if needed. This synergy between applications facilitates the communication of articulation work without disturbing production work.

Tracking Development Progress Through the Game

There are also teams where the game itself constitutes the main reference for progress tracking. The team of P7 forgoes the use of a common task manager and prefers to directly play the game to assess development progress, surpassing written tasks.

Calendars as Task Management Tools

Shared calendars can be used to share meeting times among the team. In addition to meetings, participant P4 utilizes calendars to block time for their tasks and manage them. This presents an alternative method of task tracking, which is centered around time management, particularly the duration of time required to complete a task until meetings.

Tracking Changes Through Change Log

Another mentioned alternative to task managers for tracking tasks was expressed by participant P7, whose team utilizes an internal change log to keep track of feedback and changes done to the game. With this approach, rather than tracking upcoming tasks, completed work is documented.

4.1.1.3 Task Differentiation

Within task management systems, developers differentiate tasks based on the task weight, task status, development stage, and task creator. The primary methods involve prioritizing tasks by estimated time investment, excluding quick tasks from formal tracking if inputting them takes longer than completion; visualizing task status using color-coding and tagging within shared platforms to monitor progress and facilitate feedback; and categorizing tasks by development stage. In addition, tasks are created by team leads or developers themselves, creating two distinct user scenarios for task management software. These strategies aim to create a clear, streamlined workflow that fosters shared awareness and improves overall team productivity in a collaborative game development environment.

Through Task Weight

When deciding what tasks to put on the task management tools, developers differentiate tasks based on the importance and expected time. Tasks that can be completed swiftly are excluded from task management tools and taken care of immediately, especially if the task is expected to take less time than entering it into the task management tool, as expressed by P1, P5, P8, and P11. If task managers are to be utilized for the tracking of these quick tasks, the time required to input them should be reduced.

By Visualizing Task Status

Participants describe how they visualize task status to improve collaboration and efficiency. Color-coding and tagging were employed to differentiate tasks. P6 exemplifies this by describing colors that denote stages, such as “feedback required” (yellow) and “completed” (green). They utilize a shared Google Sheet to share tasks and task status, with additional notes for tasks requiring further explanation. P11 says their team is using a task management platform, ClickUp, which allows them for tracking tasks through stages (to-do, in progress, bug, done), facilitating iterative progress and error correction. The overall aim was to achieve clear task differentiation, effective feedback, and a streamlined workflow; which are important to illustrate in a tool supporting collaborative game development, improving shared awareness in a team.

Based on Development Stages

In the realm of hyper casual game development, the process involves testing the game with users prior to continuing with development. The development only continues if the testing outcomes are in range of what is needed. This creates a unique need in task management, where some tasks are only applicable if the development continues after tests. P8 and P11 mentions they keep these kinds of tasks separately. Participant P8’s team forwards improvement tasks to another team, which will be continuing with the development after the testing. A need for categorization of tasks based on production stage arises from these examples.

4.1.1.4 Personal and Team-level Management

In participants’ game development teams, tasks are either created by product manager and team lead, or the developers write down their own tasks based on what is expected from them. This creates two user scenarios for task management software.

4.1.2 Communication

Under the communication title, patterns followed for portraying ideas and channels for file sharing; strategies like synchronicity in remote communication, hierarchy and team structure, maintaining constancy and coherency, and role-specific adaptation in developer communication are examined (Table 4.2).

Table 4.2 Strategies and Dimensions of Communication

Communication	
Portraying Ideas	Using Visualization to Convey Ideas Utilizing Tools for Rapid Mockup Creation and Idea Sharing
Multiple Channels for File Sharing	
Synchronicity in Remote Communication	
Hierarchy and Team Structure	Ensuring Quality Through Oversight of Junior Work Streamlining Feedback Through the Team Lead Facilitating Problem-Solving with Direct Contact Managing Information Flow in Team Meetings
Need for Constancy and Coherency	Regular Team Oversight of Work Continuous Communication for Game Coherence Quick Communication for Effective Problem-Solving Clear Communication to Avoid Misunderstood Ideas
Role-specific Adaptation in Developer Communication	Tailoring Interactions to Developer Roles Addressing Communication Gaps Across Roles

4.1.2.1 Portraying Ideas

Within a game development team, various methods are employed to communicate ideas and decisions, considering the diverse media types present in games. Using visualization is the most common approach in this category, followed by rapid mockup creation.

Using Visualization to Convey Ideas

Various types of visualization tools are employed in the game development process, including screenshots and screen captures, sketching, the creation of visual representations (such as UI visuals with implementation notes), referencing external materials, and flowcharts. It may be expected that artists and designers would be the supporters of this approach; however, ways of visualization are mentioned by all participants, regardless of their roles in the game development process.

Screenshots and screen captures of gameplay are a common and crucial way to communicate during the game development process. They can be used to show parts of the game that require changes to be executed by other developers in the team, to point out bugs that need to be addressed, or to convey how the game looks and feels on other devices. In the interviews, participants mentioned utilizing screen capture applications that are included in their operating systems or third-party applications specialized for capturing video, based on the requirements of the task. P11 mentions using the recorder embedded in the Unity game engine, where the code they write gets tested. In addition, P3 uses a specialized form of recording software combined with video editing software as their quality assurance role requires greater attention to detail. P2, P7, P8, and P10 utilize the stock screen capture function in the operating systems. The screenshots and screen capture videos are annotated and shared with colleagues through instant messaging apps like Slack and Discord. However, the video file size is a huge concern since these apps have limits on file sizes; P10 mentions uploading their videos to YouTube (the video-sharing platform) or WeTransfer (the file-sharing platform). As these examples demonstrate, developers

may need specialized software to capture the gameplay and annotate it, enabling them to share it with their teams without hitting a size barrier.

Participants emphasize the significance of sketching as a crucial method for conveying ideas; which is also supported by Huang et al. (2024), their work contends that sketching can be effectively employed to describe intricate tasks. P4, P9, and P11, who are participants with different backgrounds, mentioned sketching. However, particularly while working with developers not from an art or design background, the difficulty of conveying complex ideas verbally is emphasized, and sketching is used to overcome this communication gap by bridging differing mental models.

Creating detailed visuals like placeholders, collages, and prototypes with implementation instructions are also utilized in game development communication.

P1 mentions using placeholders helps developers and artists work simultaneously, without ending up with incompatible code and game art. Especially in game UI, visuals representing how the elements should be placed with placement details are created by designers for programmers. These visuals can also detail interactions and corresponding animations. Similarly, flowcharts are also used to explain game interactions and story flows. Almeida and da Silva argue that diagrams and storyboards can ease expressing game events and character behaviors (Almeida & da Silva, 2013), which is also supported by the interviewees.

As mentioned earlier, collages made with references to various media are also utilized. P7 suggests them as an alternative to sketching: "*The programmer says I can't draw. OK, then make a collage. Here I took Rollic's game X. I took this character from there. Here I took the video of this guy I saw on TikTok. I put it in the corner. I'm thinking of something like this. I took 4 UI buttons from there. I placed them like a collage. You know, I copied and pasted.*"

Based on these findings, a tool to be used in game development should implement diverse visualization tools and methods for idea sharing, such as annotations on

screenshots, video captures for gameplay, and representative visuals with detailed instructions. Flowcharts and collages are notable examples to be used as means of visualization in hypercasual game development.

Utilizing Tools for Rapid Mockup Creation and Idea Sharing

Participants express a need for prototyping tools that facilitate rapid ideation for visual concepts and game mechanics. A tool integrating drag-and-drop mechanics for quickly trying different game mechanics is desired by P1, P3, and P5. It is important to note that these participants are not programmers; however, P8 and P10, participants with the programmer role, mention that they write code for alternative mechanics and remove the unused code afterwards. With most hypercasual games sharing similar mechanics and gameplay, quick prototyping using the most common game mechanics is important.

Along with prototypes representing game mechanics, P1 and P7 also suggest a similar tool can be created for visual elements to be used by non-artist and non-designer roles. P10, a programmer, narrates how alternative game mechanics are currently tried in their workflow: "*I was writing both control mechanisms in the same script, in the same file. I was doing an on-off system, or I was activating one and turning off the other.*"

These two examples demonstrate the necessity of tools that enable developers to create representative materials for game elements that are the responsibility of other roles.

4.1.2.2 Multiple Channels for File Sharing

Throughout the game development process, a constant need for file sharing is present. Developers choose different approaches to fulfil this need: instant messaging apps, physical storage devices, and version control systems. Communication apps that offer voice calls and instant messaging like Discord and Slack are also used for quick file transfers, especially during remote work. Various

platforms offering instant messaging like Telegram, WhatsApp, and email were utilized for smaller, less critical files, showcasing a pragmatic approach to communication tailored to the specific needs of each task.

Sending files and links through communication applications presents challenges for developers. The flow of messages can make it difficult for developers to locate files later on. However, the convenience of quick sharing is still desired. Another issue is also mentioned in “using visualization to convey ideas”: larger files are not possible to share with instant messaging.

Version control systems are also used to transfer files among the team. This is mostly mentioned by programmers like P10 and P11. However, P6 disagrees with the use of version control systems with every member of the team, drawing attention to the proficiency of developers in these systems: “*So we were going to open a separate branch under level design... We were going to start sending each other levels from there, but... It wasn't very productive. Because no one in the team was used to it, because no one had used it before, there was only one person who used it in his old company, we couldn't do it properly.*” P6 also says that they had to get help from programmers for each commit, which is a term representing sending within version control systems, making file sharing slower instead of faster.

In addition to digital methods of file transfer, P2, a member of one of the smallest teams in the study, mentions that their team utilizes USB memory sticks to share files among the team members. This can be attributed to working with a small team and working in proximity, allowing file sharing with physical devices.

Having demonstrated the necessity of file transfers and the methods employed by the participants, it can be concluded that there is a need for an overhaul of file sharing systems, incorporating communication tools and version control systems.

4.1.2.3 Synchronicity in Remote Communication

Synchronicity is used to refer to simultaneous work done by developers in the team. P5, P9, and P11 draw attention to the synchronicity issue of remote communication compared to in-person work. P5 and P9 are hybrid employees, and they discuss how in-person and remote work compare for resolving complex issues. With reference to remote work, they highlight the loss of spontaneous, quick problem-solving enabled by face-to-face interaction, noting that even screen sharing or remote desktop access can't match the immediacy and effectiveness of in-person collaboration.

The results show the importance of quickly getting together as it would be done in an office; communication during remote work constitutes another area of improvement in game development.

4.1.2.4 Hierarchy and Team Structure

Under the hierarchy and team structure category, there emerge four themes emphasizing the importance of junior developers showing their work to seniors for review, streamlining information flow through team leads and project managers to avoid unnecessary distractions, facilitating problem-solving with direct contact, and managing information flow in team meetings.

Ensuring Quality Through Oversight of Junior Work

In teams where multiple developers assume the same role, a hierarchical structure is utilized. Participants have mentioned that junior roles get their work inspected by their superiors. It is noted that not all of their work is reviewed, and the ones that require assistance for the junior staff are reviewed periodically. P7 mentions artists bringing their work to a senior artist to get feedback, and P11 talks about the lead programmer reviewing their code when necessary. What is used during this review and how it is placed in the game development process is not mentioned; however,

reviewing the work could be an important dimension in tools that facilitate communication in the game development process.

Streamlining Feedback Through the Team Lead

Team hierarchy is also mentioned as a key theme when talking about the feedback. P4 mentions that the feedback provided on the work is forwarded to the team lead, who turns them into actionable items. The team lead examines the feedback, evaluates its necessity, and assigns tasks solely to the related developers, which keeps other developers focused without disruptions. This also relates to task assignments discussed in task and project management.

Facilitating Problem-Solving with Direct Contact

When an issue arises, a different approach is taken in P8 and P11's teams; immediate action is taken either with a voice call or instant messaging. This also relates to the need for quick response during remote work.

Managing Information Flow in Team Meetings

The last theme in this category is managing information flow in team meetings. In meetings with missing members, the lack of documentation results in questioning of decisions made in the meeting. P6 and P8 both give an example of these situations happening in their teams. People who aren't present in the meetings are not aware of the decision-making process, leading them to propose ideas that have already been eliminated during those meetings. Tools that allow for documentation of game decision communication could be a response to this problem.

4.1.2.5 Need for Constancy and Coherency

Regular team oversight involved frequent meetings and daily updates to track progress and provide feedback. Continuous communication for game coherence was maintained through real-time visual collaboration tools like Duet, minimizing inconsistencies, especially regarding color and style. Quick communication for

effective problem-solving utilized tools like Jira, Dropbox, and Telegram for rapid feedback and issue resolution.

Regular Team Oversight of Work

In teams utilizing Agile and Scrum methodologies, daily meetings bring developers together to assess work progress. In addition to daily scrums, additional meetings are held to check progress, get feedback, and report to managers. P5 describes their end-of-day meetings: "*This end-of-day (meeting) is something that happens at the end of every day where everybody is looking at each other's work. It goes like this, we all take turns sharing screens, and if we are in the office (today was our remote day), it's something where we all get together and go desk to desk if we are in the office. I think it's very useful. We can see what everyone is doing. We can give feedback to each other.*" In other teams, similar meetings are held at different intervals, such as a few times per week or once per week. In addition to getting feedback from other developers, project managers, and team leads look at the progress of the work.

Meetings help to create awareness and are crucial for cooperative work. Since these types of meetings necessitate a different approach to work sharing, a communication feature in a game development tool can be designed to cater to this need.

Continuous Communication for Game Coherence

Continuous communication plays a great role in maintaining coherence in game development. Tools that enable screen-sharing and control allow developers to view team members' screens on their devices, effectively separating display differences from game-related elements and allowing continuous communication between developers' work to keep coherence. P9's team utilizes a tool called Duet App to share their work among the team while working on them. Duet App is primarily designed to extend a computer's screen to another display such as an iPad; however, supporting low latency and high resolution makes it suitable for a game development team to share screens to keep their work coherent.

Quick Communication for Effective Problem-Solving

The participants underscore the significance of quick communication, and they expect it to keep the workflow efficient. P1, P8, P9, and P10 mentioned instances of spontaneous communication, either facilitated by being in the office or being on a voice call constantly.

Clear Communication to Avoid Misunderstood Ideas

Another aspect of maintaining work cohesion is effective communication among developers to prevent misunderstandings that can result in excessive workload and frustration. Participants emphasize the need for the precise communication of design decisions and consistent tracking of tasks, citing instances where poor communication between art and development teams led to significant problems, such as mismatched assets and wasted resources. P1 and P6 experienced a similar situation where developers interpreted the shared document differently, leading to a mismatch between their expected and finished work. To tackle these issues, various strategies are employed by developers, such as detailed documentation, frequent check-ins, and using project management tools. P1 and P6 experienced a similar situation where developers interpreted the shared document differently, leading to a mismatch between their expected and finished work.

4.1.2.6 Role-specific Adaptation in Developer Communication

In the interviews, participants were asked about how they facilitate communication with different departments like art, design, programming, and story. Two approaches were revealed by participants: they tailored their interaction according to developer roles and they had little to no communication with some roles.

Tailoring Interactions to Developer Roles

While talking with developers with separate roles than theirs, participants state that they take a different communication approach compared to developers with the same

role. P3, P8, P10, and P11, notably current and previous programmers, indicate they use a technical-heavy language while conversing with programmers. P1 and P9, who share similar backgrounds, also observed that developers with different roles focus on different aspects of the work while giving feedback. In the example given by P1, while asking for feedback for an animation they had made, developers were commenting on the color of the model, which was not done yet. An opportunity for a feedback system arises here.

Addressing Communication Gaps Across Roles

P2 mentioned that they are communicating less with developers with programmer roles. They stated that they only transfer files to them when the art assets are finished.

Table 4.3 Strategies and Dimensions of Collaboration

Collaboration	
Improving Cross-Role Awareness	Directing Attention to Details Across Different Roles Encouraging Developers to Understand Other Departments' Work
Collaborative Decision Making	Fostering Team-Wide Idea Contribution Refining Game Ideas Through Designer Expertise
Tackling Density of Information	
Project Planning Aligned with Developers' Abilities	
Game Engine as the Central Hub for All Outputs	

4.1.3 Collaboration

Game development is a collaborative process; hence, one of the key themes found in the interviews is collaboration. The interdisciplinary nature of the game development leads to a need to improve cross-role awareness, and its collaborative nature necessitates collaborative decision-making methods. As part of collaboration,

meetings are mentioned, and their information density is examined to improve such meetings. Project planning relating to developers' abilities looks at project management through the collaboration aspect. Finally, how game engines add to collaboration is examined (Table 4.3).

4.1.3.1 Improving Cross-Role Awareness

In collaborative work combining various departments like game development, team members might have a hard time understanding what members with other roles will be paying attention to. Within this title, two strategies are examined: encouraging attention to details across different roles and encouraging developers to understand other departments' work.

Directing Attention to Details Across Different Roles

Interviews revealed that the designers meticulously handle the screen layout themselves, believing this ensures accurate implementation by the programmers and eliminates misinterpretations. It is important to provide developers with complete information, including visual representations of screen elements with detailed implementation instructions, to facilitate accurate execution. The underlying theme is the necessity of clear communication and comprehensive deliverables to streamline the development process and minimize errors stemming from misinterpretations between roles.

P1 mentions placing UI elements in the game themselves to ensure their work gets implemented the way it is intended. To achieve the same result, P9 chooses to create UI element placement in Adobe XD, which is a UI prototyping software, which creates an implementation guide automatically for programmers. UI prototyping software is specialized for this purpose; however, the game context was not considered during the development of such apps.

Encouraging Developers to Understand Other Departments' Work

To enhance cross-functional awareness, the former employer of P3 conducted an interdepartmental communication workshop. Although these types of workshops do not directly point out an opportunity for a game development communication tool, they emerge as appropriate strategies to improve communication between departments.

4.1.3.2 Collaborative Decision Making

Game development ideation brings many developers with different roles, including marketing and publishing, together. As hypercasual game development includes creating games frequently, a need for new ideas is always present, in addition to utilizing the know-how brought by expert developers to improve ideas and reach a successful game. Fostering team-wide idea contribution and refining game ideas through designer expertise are examined within this title.

Fostering Team-Wide Idea Contribution

Team-wide idea contribution is central, with all developers — programmers, artists, and designers— freely sharing and discussing ideas in regular meetings, fostering a collaborative atmosphere. Game design decisions are made throughout the game development process, from ideation to improvement made after publishing. Collaborative ideation, while sometimes leading to setbacks, frequently results in innovative solutions and a stronger sense of ownership among the developers. While data-driven decisions are valued, open discussion is prioritized to find a balance between experienced and novice developers.

P2, P10, and P11 bring attention to how important it is for all developers to participate in ideation for game design. Experienced developers bring proven working mechanics to games, while new developers are more open to trying new mechanics.

P1 notes that team-wide participation makes developers more connected to the game project, resulting in a better output. This finding is also supported by the literature; increased ownership leads to increased motivation and passion (Chandler, 2013).

It should be noted that a pitfall for developers to avoid would be getting too attached to their ideas, as mentioned by P5. They may not want to remove their ideas from the game even when it would improve the game or keep trying to implement a decision, spending way too much time, disrupting the development process.

As it is shown that team-wide collaboration leads to a better end result, game development collaboration could be supported by tools that allow developers to ideate together along the development process.

Refining Game Ideas Through Designer Expertise

Another approach to collaborative ideation is to create ideas together and then follow up with the game designer to filter the ideas generated by the team. Initially, a brainstorming session generates numerous ideas, which are then vetted by game designers to assess viability. In some cases, narrated by P5 and P8, these ideas are then taken to marketing specialists and publisher representatives, to incorporate their opinions and get help choosing a game idea for upcoming development projects. This process facilitates asynchronous ideation with multiple parties; which can be improved with communication and documentation tools. There lies a design opportunity in game idea documentation systems that maintain versions of the ideas as it is sent through developers and publishers.

4.1.3.3 Tackling Density of Information

Many types of meetings are mentioned during interviews; one particular type, which is ideation meetings, requires different necessities to be conducted smoothly. In ideation meetings, many game ideas are reviewed to be chosen for the next project, which makes them a paramount step in starting a new game development project. P7 mentions that 60 game ideas were presented in a one-hour meeting, giving each idea

only one minute. It can be inferred that ideation meetings are information-dense, and it is possible to miss important details or whole ideas if the focus is lost. In line with this, P5 states that people in meetings get easily distracted, by looking at their phones for example. They suggest increasing the interactivity of meeting tools in order to keep the attention of participants.

4.1.3.4 Project Planning Aligned with Developers' Abilities

As stated before, game development cycle starts with project planning. After an idea is chosen for development, game designer starts to detail the idea, with contributions from the team or only working with project manager. In both cases, the project manager consults with the developers to determine their capabilities and estimate the time required to complete the task. Designers and managers frequently lack a comprehensive understanding of developers' technical capabilities and constraints, resulting in unrealistic expectations and wasted time. Effective communication and a collaborative spirit are shown to be crucial to overcoming these obstacles.

Participants P1, P8, and P11 describe the process as the project manager or game designer inquires about the feasibility of the task, opinions on the decision, and the estimated time required to complete it. P11 notes that they are presented with a game design document and could ask their questions about the document to the designer in a meeting, which is followed by them writing the requirements of the project. They then work over this document with the project manager to create a project plan. It is also mentioned that this is also done by the game artist, and the production phase begins.

This stage of the game development process, preproduction, demands extensive coordination and communication between developers, game designers, and project managers. A tool to be designed to support this process would have to consider the iterative process of creating the game design document and allow project managers to combine feedback from various developers.

4.1.3.5 Game Engine as the Central Hub for All Outputs

All three programmers mention the game engine (Unity 3D) as the main software for combining the deliverables prepared by individual developers. Artists and designers further mention that they give the assets they have created to programmers instead of directly implementing them to the game. Therefore, enhancing engines by including methods for asset collection can be considered an opportunity.

Table 4.4 Strategies and Dimensions of Documentation

Documentation	
Aspects of Documentation Tools	Need for Multimedia in Documentation Tools
	Utilizing Office Applications for Documentation
	Using Infinite Canvas Tools in Documentation
	The Need for Synchronicity in Documentation Tools
Creating a Library	Collecting Assets for Use Across Projects
	Games and Media Reference Library
	Task List Drafts for Consistent Project Planning
	Personal Documentation for Self-Improvement

4.1.4 Documentation

Documentation is a fundamental aspect of game development, as highlighted by participants in the interviews. They emphasize the detrimental effects of inadequate documentation, which results in confusion, wasted time, and inconsistent decision-making by exemplifying how forgotten or poorly communicated decisions previously led to substantial delays and inefficiencies. A need for a centralized, persistent record of all project decisions and design choices is inferred by the participants. To illustrate, using document depositories like Confluence and Notion

ensures team members have a shared understanding and a readily available reference point throughout the development process (Table 4.4).

4.1.4.1 Aspects of Documentation Tools

For documentation tools, the participants' comments revolve around the need for flexible, multimedia-rich documentation tools to effectively handle the complexities of game development. It can be inferred that there is a desire for a unified platform capable of seamlessly integrating different media types, thus improving workflow and overall project management.

Need for Multimedia in Documentation Tools

In previous sections, the importance of visualization in communication was discussed. Participants also highlighted the need to document the multimedia created during the communication process for future reference in the game development.

Based on the participants' ways of documenting the process, three main points can be made for the utilization of the documentation tools: Different applications excel in communicating distinct information, necessitating the use of distinct tools in areas where the primary application falls short, since no single application fully meets all requirements, and the links from various applications are added as necessary.

Participants P5, P6, and P9 experimented with multiple applications, including Miro, Notion, and Figma, leveraging their individual strengths to manage various aspects of the project, such as flowcharts, mood boards, and wireframes, using the integration and linking features of the apps.

Utilizing Office Applications for Documentation

In contrast to the upcoming title, office applications such as word processing software (Google Docs), spreadsheet software (Microsoft Excel, Google Sheets), presentation software (Google Slides), project management software (Confluence), and note-taking software (Notion) are utilized for documentation purposes in the

game development process. These tools have a linear content flow compared to whiteboard tools like Miro and Mural. However, they offer distinct advantages in documentation: structured page layouts, user-friendly for all departments, and the ability to present comprehensive information.

Participants P6 and P8 highlight the inadequacy of single-purpose tools like Google Sheets and Confluence for managing diverse documentation needs, particularly regarding the integration of visual and textual information.

Using Infinite Canvas Tools in Documentation

Infinite canvas or digital whiteboard tools like Miro and Mural are mentioned by the participants. These tools are chosen for their visual and creative capabilities, allowing for the integration of diverse elements such as links, references, images, GIFs, and videos all within a single document.

The Need for Synchronicity in Documentation Tools

In a collaborative workflow of game development, shared digital documents have a crucial role in facilitating communication and efficient decision-making. Tools like Google Sheets, Confluence, and Miro are highlighted as integral to this process, enabling real-time feedback, streamlined approvals, and efficient documentation. This need is present in each stage of the game development process, from initial design discussions to publishing with its “living document” quality.

P6 emphasizes how using Google Sheets for data sharing and task management streamlined their process, automating and enhancing efficiency. Additionally, they have commenced utilizing Confluence for the documentation and sharing of design decisions. Furthermore, they have implemented a feedback mechanism through Confluence, positioning this approach as their preferred and effective method.

Applications that allow synchronization between collaborators are also used by participants; Google Docs, Notion, and Miro can be given as examples.

In their interview, participant P4 highlighted the "living" aspect of the "game design document": "*At the very beginning, it is actually written even before the project starts. But this is a living document. There may be a feature that will come in a new update or maybe there may be an error in the system, with mechanics. This can also be edited. In fact, this document is constantly changing.*"

A tool with synchronization capability can aid the creation of game design documents and other documents that get created during the game development process, so that these documents can be a "living document". It may be possible to utilize cloud-based technologies to that end.

4.1.4.2 Creating a Library

In the interviews, in addition to how the documentation is done, what is documented is also examined. Four areas of collections were inferred: game assets, references, repeating tasks, and personal knowledge. Motivations to collect and archive them are discussed.

Collecting Assets for Use Across Projects

Efficient asset management is important, especially in hypercasual games, where assets can be transferred to upcoming projects. Collecting and reusing assets not only saves time and resources but also helps maintain visual coherence. In addition to the assets created by the development team themselves, asset stores like Unity Asset Store are also utilized to acquire assets for current and upcoming projects.

Participant P2 asserts that since the UIs of hypercasual games are almost the same, they can reuse the UI flow prototypes in other projects with little change.

Collecting assets in a central location also aids designers and artists in referencing while creating new assets, ensuring consistency in artistic styles. When a new UI element is introduced to the game, it should be consistent with existing elements, regardless of the artist or designer, as highlighted by participant P1. They also add

that the inconsistency is only realized when the assets are implemented in the game, which makes a central collection of assets a must to keep cohesion. A similar issue is also encountered by P9's team; they have started an asset pool to create a design language for the games created by their company.

Games and Media Reference Library

Participants also underline the importance of readily accessible referencing libraries, allowing developers to easily find appropriate references to help describe their ideas. While other hypercasual games are the most prominent type of reference used, it is not limited to them; other types of games, including mobile and computer games, movies, applications, and art, can be used to reference game ideas. In addition to ideas, participant P2 argues that information about games and screenshots from them should also be included in such libraries.

Having mentioned the reference libraries, a platform where these references can be combined to make mood boards and game design documents is also a need of the participants.

Task List Drafts for Consistent Project Planning

The documentation does not only apply to creating a library of assets but also to the game development tasks themselves. The primary objective of task list drafts is to prevent repetitive task planning by establishing pre-made checklists for recurring tasks to improve efficiency and reduce errors by ensuring all necessary steps are consistently completed in each game project, eliminating the risk of forgetting crucial actions.

Personal Documentation for Self-Improvement

Participants P2 and P10 mention utilizing both physical notebooks and digital tools like Microsoft To Do to track their tasks and design process, documenting game mechanics, information architecture, and daily progress, which allows direct comparison of design iterations, enabling self-assessment and iterative refinement.

This documentation serves as a valuable resource for tracking personal growth and quickly referencing past solutions to recurring problems.

4.1.5 Tool Specific Considerations

The last title examines needs for improved tools and workflows for game development communication. Key themes include the need for integrated development tools, importance of interoperability between various applications for streamlined workflows, impact of audio and screensharing quality on communication, usability and suitability of the tools for game development and workplace professionalism (Table 4.5).

Table 4.5 Strategies and Dimensions of Tool Specific Considerations

Tool Specific Considerations	
Integration between Development Tools	Need for Previewing Various File Types in Development Ensuring Effective Application Integration for Streamlined Workflows Integrating AI Applications
Cooperative Work Through App Features	Centralizing Articulation Work for Efficiency Sharing Developer Status Updates for Better Coordination
Impact of Communication Tool Quality	Limitations of Screen Sharing Impact of Audio Quality on Remote Work
Ensuring Usability of Shared Applications for All Developers	Version Control Systems Used by All Developers Usability of Tools for Articulation Work
Aligning Application with Work Environment	

4.1.5.1 Integration between Development Tools

The interviews revealed that there is a crucial need for better integration between development tools in game creation. First topic explored in relation to this integration is the necessity of improving asset previewing and code sharing, highlighting diverse file types and the preference for centralized file sharing within communication platforms. Secondly, various platforms allowing integrations are discussed to show how they add to streamlining workflows. Lastly, the integration of AI tools into game development was explored to improve game development.

Need for Previewing Various File Types in Development

Inferred from interviews, there is a critical need for improved asset previewing and code sharing within a game development workflow. There is a lack of a universal preview program for various types of file formats used by developers with different roles. These file types may involve 3D models, images, texture files, and other game engine materials. The need for previewing various file types also applies to file sharing and communication systems, since participants prefer to share files through communication apps as well. Therefore, the participants assert the demand for a centralized system enabling seamless previewing of diverse file formats and a more integrated code-sharing mechanism, ideally within a team communication platform, to streamline the development process and improve team communication.

Ensuring Effective Application Integration for Streamlined Workflows

Participants highlight the importance of seamless interoperability between different apps and platforms, emphasizing the need for user-friendly design that minimizes disruptions. The discussion centers on the advantages and disadvantages of various platforms like Google Workspace (Calendar, Teams, Drive), Slack, Trello, Jira, and Adobe Creative Suite, comparing their integration capabilities and impact on workflow efficiency.

P5 finds Google Workspace, a comprehensive suite of productivity tools provided by Google, to be a logical and efficient solution. This preference stems from its seamless integration with the participant's existing applications, including chat, calendar, video conferencing, and cloud file storage. They also mention using Slack, Trello, and Jira together in the past for this purpose; these apps also offer integrations without being made by the same developer. Similarly, P11's team utilizes Microsoft Teams due to the seamless integration of scheduling meetings on a calendar and joining them.

P3 and P4 highlight the usefulness of synchronization between platforms. P3 states that starting a task on one device and finishing on the other is a benefit of such synchronization, which is also favored by P4 when notified by Google Calendar on the phone when a meeting is soon. Another integration example was given by participant P9; they utilize Adobe Creative Cloud applications like Illustrator and Adobe XD, which allows working on the shared files. These examples demonstrate potential integrations between tools utilized in game development. Integration within communication apps and tools aiming to organize articulation work is particularly important in game development communication.

Integrating AI Applications

Integration of Artificial Intelligence (AI), specifically generative AI and Large Language Models (LLMs), into game design and development is also a concept brought up by participants. The main focus of AI integration is the fast creation of ideas and visuals to help with game ideation. Both P2 and P7 suggest that LLM integration within game development tools can help with overcoming problems in ideation. In addition, quick visual creation supported by AI applications is found to be helpful in conveying game ideas. It is expected that at some point in the future, these tools also will be used to create assets that will be implemented in the game. Another area of game development AI tools that could be implemented is mentioned by P8; they prefer to use CoPilot and ChatGPT premium features to help with their coding.

With these examples, it can be argued that AI and LLM tools could be implemented in applications, especially to help with conveying ideas through quickly creating visuals and alternatives during game design communication.

4.1.5.2 Cooperative Work Through App Features

This theme focuses on how tools used in game development for task management, communication, and file sharing support articulation work, especially coordination within the team.

Centralizing Articulation Work for Efficiency

During interviews, it has become evident that a centralized communication and task management platform is required to aid the teamwork during game development . Participants express frustration with using multiple applications, like Slack, Teams, Trello, and Discord, for different aspects of work, advocating for a single platform that integrates task management, communication, file sharing, and performance metrics.

P7 and P8 describe an umbrella application that integrates task management, documentation, file sharing, and chat capabilities into a cohesive platform.

In participant P11's team, both Slack and Microsoft Teams, applications that address the same need, are utilized. One application is mandated by the company, while the other is employed within the team due to the mandated one's limitations in fulfilling all their requirements.

The current fragmented system is inefficient and presents an opportunity to create a more streamlined workflow, incorporating features of existing tools into one unified solution.

Sharing Developer Status Updates for Better Coordination

Maintaining focus during work is paramount, with developers using applications like Slack and ClickUp to signal their availability, including breaks like lunch. P8

highlights that they detest being asked about the task status often, especially while working on the said task, stating it makes them feel inadequate. Their team also values spontaneous meetings, which happen when people are chatting on Discord and decide to jump on a voice call to discuss, which requires knowing who is available. This connects with the need for awareness in collaborative work; developers being aware of their colleagues' availability improves communication during game development.

4.1.5.3 Impact of Communication Tool Quality

In line with the findings from the literature, participants also brought attention to the quality of communication the communication tools offer. It is highlighted that low-quality screensharing hinders the ability to effectively demonstrate intricate details crucial in game development and the importance of high-quality audio for successful remote collaboration during game development.

Limitations of Screen Sharing

The limitations of screen sharing as a communication tool, particularly within the context of game development and remote work, are also discussed during the interviews. In video games, errors that occur in split seconds or result in a disconnect between input and the game's output are frequently identified and corrected. However, participant P11 contends that low-quality and low-framerate screen sharing is insufficient for demonstrating such intricate details. Trying to overcome this, participant P10 mentioned that they had to upload screen capture videos to YouTube to show the details that are not visible with the screen sharing offered by communication apps.

These aspects of game development necessitate the adoption of superior screen sharing systems compared to those currently available in general communication applications that lack specialized functionality for game development purposes.

Impact of Audio Quality on Remote Work

As also stated in the literature by Olson and Olson (2012), audio quality is critical while working remotely. Participant P8 had a negative experience with a platform's audio and screen-sharing capabilities. They further praise Discord's superior video compression and real-time noise cancellation. This underscores the significant impact of these technical features on remote collaboration. It is important to note that Discord was primarily designed for game screensharing, not for development purposes; however, it compares favorably in meeting the needs of the game development process.

4.1.5.4 Ensuring Usability of Shared Applications for All Developers

In the interviews, the usability of shared applications is highlighted by participants, and why some applications were found unsuitable was examined. Two primary types of tools were prominent: version control systems like GitHub, SourceTree; and task and project tools like Trello, ClickUp, and Jira.

Version Control Systems Used by All Developers

Among the various tools utilized during game development, there are some specific tools targeting the needs of a particular department. However, since the game development process brings various departments together, tools like version control systems (usually designed for software development) are used by all developers regardless of their role.

In P6 and P7's team, version control systems are used to collect all elements of game development together. P6 mentions that they were asked to send the level design files through the version control system; however, since the level design team had lacked programming background, they encountered problems, and the use of such systems was only left to programmers. Similarly, in P7's team, all developers are expected to use version control systems; unlike P6's team, they are able to implement the use of the systems throughout the team regardless of their programming knowledge.

When version control systems are only left to programmers to use, developers in the team need to rely on them to implement every element into the game, as mentioned previously by P6. This results in heightened workload for programmers and frustrations in communication due to the workload shift.

While utilizing version control systems helps with bringing game elements together and facilitates an effective collaboration, problems with its use by non-programming roles are hindering the overall acceptance of these tools. Game development communication could be improved with designing tools that help bring game elements together without relying on programming knowledge.

Usability of Tools for Articulation Work

Task and project management tools are expected to be used by every role in the game development process. There are many applications offering different levels of detail, and a suitable one should be chosen for the process to avoid extra mental load. Both P4 and P11 highlight this by comparing Trello with other task management tools. Trello is a digital tool that emulates a kanban board, where users can write tasks in a board and move them around categories like "not started" or "done" to describe their state of completion. It is found to be more suitable for smaller teams with basic projects due to its simplicity, whereas ClickUp and Jira are found to be more complex. When developing hypercasual games with 3-person teams, Trello was chosen, but when transitioning to hybrid-casual games, teams opt for ClickUp and Jira.

A consideration for designing tools that support game development arises here; the scope of the tool should fit with the scope of the game development project.

4.1.5.5 Aligning Application with Work Environment

Communication and project management applications need to be appropriate for both the game development process and work environment; they should be professional in social and technical contexts.

Although games are produced for entertainment purposes, maintaining a professional attitude is still expected during the game development context. P3 and P7 express a strong preference for professional tools like Slack and Microsoft Teams over platforms like Discord, citing concerns that Discord's informal tone and gamer-centric features are unsuitable for corporate settings, regardless of its suitability for game development.

Tools like Discord and Trello fall short when teams are growing. Since Discord was designed for personal use within friend groups, limits on the number of people in voice and video calls fall short, pointed out by P8. They also compare Discord to Microsoft Teams, in which the subscription model for premium features works differently; Discord sells subscriptions for individual accounts, and Microsoft Teams' customer is the company and sells team subscriptions.

A potential communication app for game development also should consider creating a sense of team unity, especially in remote working conditions. P8 exemplifies this by saying "*For example, when we were doing a video call, after a certain number, not everyone could enter the video. I think it should have to happen, especially for a company that works remotely. ... On the one hand, people shouldn't think that they are sitting in a room alone, people should think that they are working together.*"

To sum up, tools to be designed for game development should consider these three aspects: professional appearance, team focus, and team unity.

4.2 Stage 2 Findings

In the second stage of the study, which was conducted as a generative workshop, the participants discussed the issues they encountered during game development communication and explored solutions. These solutions can directly or indirectly guide the development of new collaboration tools, as well as constitute a reference for planning the overall process.

Issues raised and solutions suggested are grouped deductively under the five categories established from interviews, with no findings falling under the topic of tool specific considerations.

4.2.1 Project and Task Management

Clear Task Management

It was stated that in the task management tools, tasks are written with insufficient detail, which requires developers to ask the team lead or project manager to learn more about their tasks. Similarly, failing to check task management tools and instead asking other developers in the team was found to be an issue during the workshop, in line with interview findings. Both issues create frustration during the communication of tasks. Task management tools should support both describing and reviewing tasks clearly.

Integration with Project Management Tools

Issues related to developers missing tasks and failing to update project management tools like Asana surfaced within the workshop. It can be inferred that there is a need for a consistent workflow where project and task management and work are brought closer.

Time Estimation, Planning, and Sprint Review

The discrepancy between time estimation of tasks and the actual time the task requires was a more prominent concern in the workshop compared to interviews. In addition, it was noted that problems with the created work are realized too late in the development, requiring reworks and delays. A need for realistic, organized development cycles and accurate scheduling arises from these issues. Periodic revisions attended by all developers are suggested by the participants; in that case, a tool facilitating such revisions could be proposed.

Knowing the abilities of the team members is also presented as a solution to project management issues. A game development tool could help this by documenting the time passed for each task and recommending a timeline for upcoming projects based on the documentation.

Failing to plan in accordance with team members' individual abilities is also reported to lead developers into acting on their own during development. Tools that help with project planning could also include features that allow team members to better convey their unique abilities to other team members, thereby helping to resolve this issue.

4.2.2 Communication

Feedback and Clear Communication Channels

During the workshop, the most frequently mentioned issue was the tendency to become overly fixated on a particular idea in game development. Several strategies are suggested: getting external feedback from players or people outside the team, focusing on the team effort, encouraging novelty, and trying to understand opposing ideas better. This creates opportunities for game development tools for communication in two ways: facilitating an external feedback system and tools that help convey ideas better for discussions.

A similar problem emerges from miscommunicated or poorly supported ideas, as stated by the participants. Only disagreeing without an alternative solution or suggesting baseless alternatives is found to be frustrating. Tools to help convey ideas with supporting elements like sketches, diagrams, and data could be utilized to help with communicating ideas during game development.

A technical issue is reported to be arising from using several communication channels. It is recommended to overcome this by using a single channel and a single application for communication. However, this could lead to missed messages, for which implementing threaded conversations could be recommended.

Constructive and Effective Communication

Problematic behaviors encountered during game development are identified by participants: lack of effective communication, hesitation to admit lack of knowledge, blame-shifting, and feeling unheard or undervalued. To address these issues, it is recommended to create an open and accepting environment that facilitates collaboration among developers. Effective communication strategies, such as the sandwich tactic where each negative feedback is delivered between positive remarks to ensure no negative feeling is accumulated during the feedback process, are suggested to promote open dialogue. Additionally, ensuring that all developers feel their input is valued is found to be important in fostering a positive and productive work environment. While these strategies are mainly achieved by behavioral change, a game development tool can support these efforts through encouraging open communication, collaboration, and transparency.

Navigating Workplace Dynamics

Relationship dynamics and interactions between colleagues and managers were also presented as causes for issues by the participants. Rigid hierarchical order, a lack of transparency from the management team, and a lack of alignment of goals and commitment between team members are identified as problems. Participants related these issues with human resources management; however, tools that help maintain a positive workplace culture could be implemented to overcome such issues.

4.2.3 Collaboration

Interdisciplinary Awareness

In the workshop, participants also underlined the gaps in communication between developers with different roles. Utilization of mechanics testing, moodboards and references, prototypes, concept work, and working with alternatives were given as solutions. It can be inferred that collecting and documenting such multimedia is a need in the game development process.

It is reported that, for quality assurance testers, proper conveying of bugs is key. Developers expect steps to recreate bugs in detail and necessary visual aids when bugs are reported. Failing to relay the necessary information creates frustrations in communication, requiring repeated conversations between the tester and the relevant person. A possible solution to this is implementing systems where bug reporting is standardized, thereby helping QA testers to convey their findings more effectively.

Overcoming Monotonous Tasks

It is stated that tasks can get tedious and monotonous. To address this issue, the following strategies are suggested: taking a break, switching tasks, and seeking input from colleagues regarding the task.

4.2.4 Documentation

Documentation of Approved and Eliminated Decisions

Project management applications facilitate task distribution but do not provide adequate options to document the decisions taken throughout the process clearly. A symptom of the same issue can be observed when the resulting product does not match the idea. To these issues, participants suggested keeping a roadmap with complete guidance and a record of eliminated ideas that will not be used. This shows a need for a system to help both approved and eliminated ideas to be documented.

Centralizing File Sharing

Both interviews and workshop studies showed that communication tools that facilitate instant messaging are used to transfer files between developers. This is found to be an issue in a few ways: communication apps don’t support large file sizes, it disrupts the information flow of communication, and it makes it hard to find the files at a later date.

To overcome file size limits, it is recommended to use portable data storage devices like memory sticks and external drives, or cloud-based file storage services, which

also creates issues. Using portable storage devices is impractical in offices and even impossible during remote work. Cloud-based file services help with issues of physical devices; however, a proper file organization is still required to utilize it effectively. Integrations between communication tools and cloud-based file systems could help with file storage and organization issues.

Another issue encountered during game development related to files is developers not following naming conventions required for implementing into game development software. If the creator of the asset and the developer implementing it are not the same individual, a frustration in communication could develop over time. A game collaboration tool to be designed could help developers to maintain naming conventions during file transfer to help with the communication issue.

Version Control and Task Awareness

As discussed previously on stage 1 findings, during game development, version control systems are utilized to share files, in addition to their primary purpose. However, infrequent pushes to the repository result in conflicts and create frustrating work for the developer trying to merge the commits. It is recommended to commit and push frequently, in addition to asking colleagues if they are also working on the element. As this relates to awareness, knowing what tasks are being worked on, creating task awareness, could be shown in project management software to overcome issues.

Bringing Information Together

Game development, especially hypercasual development, requires a specific know-how developed upon data created from the previous games published, stated by the participants. While the suggested solution was to listen to experts in the area, the issue can be taken as a documentation problem. This creates a need to incorporate testing and marketing data to be accessible while developing games, which can be accomplished by implementing a suitable documentation system within project management software.

CHAPTER 5

CONCLUSION

In this chapter, the study is summarized and concluded under four subtitles. Firstly, an overview of the research is presented, and the methodology is summarized. Secondly, the research questions are revisited, and the findings are discussed in relation to how they address those questions. Before finalizing the conclusions, limitations and further research are discussed.

5.1 Overview of the Research

The aim of this thesis was to identify areas for design intervention for communication of game design and collaboration between team members from diverse backgrounds, with a focus on hypercasual game development. Detailed aim and scope, research questions, and background of the study are found in *Chapter 1*.

Initially, the literature is reviewed to set the necessary background for the studies that will seek answers to research questions. *Chapter 2* explores the literature on collaboration with a specific focus on computer-supported cooperative work (CSCW), the concept of (video) game and game development processes, and the studies examining collaboration practices within the game development teams .

The methodology is outlined in *Chapter 3*; which provides a detailed description of the qualitative research approach followed in the present study, research stages and design, sampling approach, research conduct, and the details regarding the thematic analysis. Qualitative methods were used in the study to gather in-depth insights about game design and development communication and collaboration. The study consisted of two stages: Semi-structured interviews followed by a workshop, both conducted with game developers with different roles. The findings derived from the

analysis of the interviews and the workshop are presented in *Chapter 4*. They revealed the key themes that frame the design intervention areas for enhancing team work in the game development process through better process/task management, communication, collaboration, documentation, and effective use of game development tools.

This chapter, *Chapter 5*, summarizes the thesis, integrates the literature review and findings from the two-stage study to address the research questions, draws conclusions, and provides recommendations for further research.

5.2 Revisiting Research Questions

The literature review and data from the semi-structured interviews and workshop are analyzed to provide responses to the following research questions:

> *Q1. What content is being communicated between the members of the game design team throughout the (game) development process?*
>
> *Q2. What are the strengths and limitations of the existing tools and/or methods utilized in mobile game development for design communication and team collaboration?*
>
> *Q3. How can the communication of mobile game design and the collaboration between team members be supported by design during the game development processes?*

5.2.1 What content is being communicated between the members of the game design team throughout the (game) development process?

The semi-structured interviews revealed the initial categories, with the help of the literature.

- Team Coordination and Project Management

- Information pertaining to meetings, encompassing both the planning phase and outcomes, is communicated among the game development team.
- Task assignments are relayed to the team using verbal communication, communication tools, and task management tools. Task assignments through tools can include deadlines and progress.
- Updates on tasks and work, what task is being worked on, and availability are communicated within teams, during meetings or with the help of task management tools.
- Project planning is done with the participation of team members; developers collaborate to create timelines based on their abilities and estimated task completion times.
- Task clarification and discussion are facilitated within task management tools, communication tools, or verbally. Comments are utilized to pose questions and obtain detailed information on tasks.

- Game Ideas and Design Decisions
  - Screenshots, screen captures, sketches, and diagrams are used to communicate game ideas and design decisions. Visual placeholders, collages, and prototypes are also created to represent game ideas and design decisions.
  - Game mechanic alternatives are coded and assessed within teams to convey design decisions.
  - Feedback and reviews on each other's work are communicated verbally, through project and document repositories, or through communication tools.
  - Game ideas are created in ideation meetings and filtered by the experts afterwards.

    - Continuous communication is utilized to maintain coherence in game design and development.

- Documentation and Knowledge Sharing
    - Game design ideas and decisions are documented in document repositories to be utilized in later stages of the game development process.
    - Game design documents are kept up to date by developers as “living documents”, to communicate game design decisions throughout the development process.
    - Data provided by publishers and marketing departments is typically distributed to developers in spreadsheet format, enabling them to implement modifications to the game based on the provided information.
- File Sharing
    - Developers share various types of files, including 3D models, images, and text files, using communication apps, physical storage devices, and version control systems.
    - Developers share references to other games, movies, applications, and art to communicate their ideas.
- Bug Reporting & Feedback
    - Feedback on completed and ongoing work is communicated within the team to facilitate continuous improvement.
    - Change Logs are used to track progress on feedback and changes.
    - Quality assurance testers communicate about the steps to recreate bugs and visuals that are being reported.

5.2.2 What are the strengths and limitations of the existing tools and/or methods utilized in mobile game development for design communication and team collaboration?

During the interviews, participants mentioned a large number of tools and applications they currently use or used in the past in the game development context. This diversity implies a lack of purpose-designed tools for collaborative game development. Strengths and limitations of the tools used during game development communication can be discussed as follows:

Strengths

- Specialized Technical Capabilities: Unity, as a game engine, provides the infrastructure to create the game and offer a collaboration environment.
- Workflow and Task Tracking: Asana, ClickUp, Jira, Monday, and Trello were mentioned as project management tools that facilitate the organization of tasks, assignments, and deadlines.
- Centralized Knowledge Sharing: Confluence and Notion are mentioned to provide space for documentation and wikis.
- Real-Time Communication: Discord, Slack, Google Meet, Microsoft Teams, and Zoom are used for immediate communication for quick problem-solving and virtual meetings.
- Visual Collaboration: Miro and Mural are mentioned to be used both synchronously and asynchronously in collaboration, facilitating a platform for visual references and shared design documentation.
- Integrated Platforms: Google Workspace and Microsoft Teams offer unified platforms for communication, file storage, and project management. Jira, Trello, and Slack offer integrations between the applications to create workflows within them.

- Flexibility: Tools like Miro and Mural offer flexibility through both synchronous and asynchronous use.

Limitations

- Lack of Integration: Within a game development process, developers utilize a large variety of tools, which then requires them to switch between apps, creating a fragmented workflow, hindering communication between developers.
- Task Management Inefficiencies: Tasks entered in task management tools might lack necessary detail, requiring further elaboration or resulting in miscommunicated work. Developers might fail to check tasks or enter progress, creating a lack of task awareness.
- Project Planning Issues: Project planning may not accurately reflect developer capabilities and constraints, leading to the discovery of issues late in the development process, necessitating rework and delays.
- Documentation Challenges: Document repositories failing to handle the integration of multimedia like Confluence are not suitable for game development, since such multimedia is required to relay game design decisions. Moreover, failing to keep the documentation up to date creates communication issues.
- Usability Issues: Tools such as version control systems are specialized to be used by programmers and are beneficial to be used by all developers in the team; however, adoption of such tools into workflows is hindered by their complexity to non-programmers. Task and project management tools might get too complex for small teams with smaller scopes.
- Inadequate Communication Features: Game development context requires additional tools compared to other development processes working remotely. High-quality screen sharing with a high framerate is required to show the level of detail necessary in the game development process.

- Limitations on File Sharing: Most communication tools have file size limitations, due to being communication-first applications. In addition to file size, sharing files through communication tools makes locating files later difficult.
- Missed Information: When multiple communication applications are utilized within a team, there is a risk of messages being overlooked, and maintaining an accurate record of information becomes challenging.
- Unprofessional Appearance: Tools like Discord that lack a professional appearance are avoided regardless of their suitability in game development collaboration.

Strengths and limitations of the tools used in the game development process related to communication are inferred from the interviews; then they are analyzed in line with the recommendations given by participants for the next question.

5.2.3 How can the communication of mobile game design and the collaboration between team members be supported by design during the game development processes?

Game developers' needs and expectations relating to communication during the game development process are inferred from the interviews and the workshop. A collaboration tool designed to enhance communication among developers throughout the game development process should be tailored to meet the following requirements and expectations of developers. The needs and expectations highlight considerations for design interventions in game development collaboration tools. Considerations are categorized under the following groups: project and task management, communication, collaboration, and documentation.

Project and Task Management

Task management, notification systems, task tracking, task differentiation, and integrations could be improved if the following considerations are met: Task management tools should allow developers to monitor progress and assign tasks through features like assignments, deadlines, and comments to maintain awareness within the team. Tools to allow task differentiation based on priority, status, and development stage should be implemented in task management apps. Task managers should support writing tasks with necessary detail, in addition to displaying tasks clearly to avoid miscommunicated tasks. Notification systems should balance efficient task assignment with minimal interruptions; integrations with other systems could improve task communication. Tracking task progress requires alternative ways other than manual reporting. Documenting time spent on each task could help with project timeline estimations. Integration between task management tools and project management tools is important to prevent missed tasks or frustrations based on disrupted workflows.

Communication

Visualization tools, rapid prototyping, file sharing, synchronous communication, and feedback support are areas of intervention to improve communication in the game development process. Visualization and multimedia tools are important in game design communication as they support idea sharing; sketches, diagrams, and data help with conveying initial ideas, and screenshot annotations, screen capture videos, representative visuals, collages, and flowcharts further help with design communication during the game development process. Rapid prototyping of visual concepts and game mechanics is an expected feature in game development communication supporting tools. File sharing should be centralized and supported with integrations to communication tools to facilitate transferring files rapidly and accessing them easily later. Immediate communication should be supported to quickly respond to issues, similar to in-person interaction in an office to improve remote work. Tools should facilitate documentation of meetings to log game design

decisions for team members who are not in the meetings. Additionally, these tools should provide a means of referencing notes for future reference. Threaded conversations could be implemented in communication tools to organize communication and centralize it to a single application to avoid missed messages. Feedback and bug reporting should be streamlined in a tool, and it should include visuals and guidelines to recreate the issues. The tool could potentially integrate AI tools for the fast creation of ideas and visuals. Communication tools utilized for game development should incorporate optimized screen-sharing and audio calling capabilities to facilitate high-quality and high-framerate screen-sharing.

Collaboration

Cross-role awareness, team-wide idea collaboration, interactivity in meetings, and project plan support could be improved to make design collaboration better. Cross-role awareness can be improved by implementing tools that help with rapid prototyping, which can be used by every developer regardless of their roles. Tools can help with collaborative ideation to facilitate team-wide idea contribution. Interactive meeting support tools could help developers stay focused during meetings. Tools that help project managers create accurate timelines by providing knowledge about developers' abilities are required.

Documentation

Flexibility, unification, and synchronization are important for documentation tools in game design communication. Libraries of several elements could be implemented. Integrating various media types for flexible, multimedia-focused documentation could help with documenting game design decisions. It is crucial to centralize game design documentation in a unified platform that is capable of accommodating a wide range of documentation requirements, both in structured layouts and infinite boards. Synchronization of documentation is needed for real-time collaboration, in addition to creating "living documents", that is to be changed alongside the project process. A specialized tool is required to facilitate the creation of reference libraries in game development documentation. This tool should enable the seamless integration and

previewing of diverse file types, including images, 3D models, and code snippets, facilitating effective communication among developers. Collecting assets to reuse and creating task list templates to be used in each project should also be supported. Making testing and marketing data more accessible to developers during the game development process should also be considered for such a tool.

5.3 Future Research

This thesis focused on the development process of hypercasual games while examining the communication between developers with various roles. An important aspect of hypercasual games is the duration of their development time, which changes from one week to three months, as reported by the participants. Examining game development projects with longer time scales may yield further insight into the topics this thesis studies.

While focusing on communication between developers, the study revealed insights into the communication of different roles without looking deeper into particular role pairings, like designer-programmer communication or artist-designer communication. Studies on such pairings could yield deeper insights into game development communication and collaboration.

Alternative to the methodology used in this study, a longitudinal approach could be taken to understand the collaboration within a game development team, divided by different stages of the game development process.

5.4 Final Remarks

By conducting interviews and a generative workshop, it was possible to attain a deeper understanding of teamwork during game development.

The first inquiry of this thesis related to the content of communication between team members. This was found to be mostly tasks and feedback related to the game being

developed. The nature and the volume of communication depended on the roles of participants and other team members they were communicating with. In particular, the difference between within the same role and between diverse roles communication was noted.

The study also inquired into the strengths and limitations of the tools used in the industry. Participants emphasised the lack of tools designed for game development purposes and their reliance on tools designed for different areas. This can be attributed to the new emergence of the hypercasual game industry.

The final aim of the study was determining ways to improve communication of hypercasual game design. In that regard, most participants focused on tools that made prototyping easier and a general reference library accessible by all team members. The importance of the ease of prototyping is important when breaching the terminology gap created by the interdisciplinary nature of the work. By reaching a prototype faster, team members can better align their understanding of the project and share their feedback. The significance of the reference library is more related to the standardization of game development and ease of access for resources.

However, it was also observed that while participants are fast to point out problems they faced during teamwork and they can agree on several solutions, they do not consider their problems dire enough to actively try to find better tools. This can also be attributed to the lack of available alternatives for hypercasual game development collaboration tools. This can be considered as the most significant finding of this study as it not only shows that a need exists for enhancing teamwork during game development, but also the fact that this need is not currently met by the available solutions.

Communication in game design and development is affected by many aspects. To improve this communication, it is important to understand the game development process and spot frustration points that can be improved by refining communication between developers, especially those with various backgrounds. In this study, the complex landscape of collaboration and communication within game development

teams is explored in relation to the dynamics between developers with various roles, revealing areas where gaps occur in communication. Although existing tools possess strengths in specific domains such as task management, communication, and file sharing, their limitations impede efficiency due to the absence of integration, inadequate communication capabilities, usability concerns, and the inability to effectively manage multimedia content.

The study highlights the importance of considering role-specific needs and the impact of team dynamics on the success of communication and collaboration while also emphasizing the necessity for considerations in game development tools to facilitate clear communication of tasks, support idea sharing through multimedia elements, create awareness in project management within developers, and create a guideline for effective documentation. Ultimately, the findings underscore the need for a tailored solution for communication in the hypercasual game development process, with its specialized needs during the development process.

The focus of the study is the hypercasual game industry. However, the results found can also be applied to the game development industry in general. The literature related to interdisciplinary teamwork in game development tends to focus on programmers' or game artists' perspectives. As this study did not focus on any particular role, its findings may offer novel insights for that literature. Finally, the participants worked in teams that follow software development methodologies like agile and sprint, which implies a certain applicability of the results found here to other software development work with interdisciplinary teams including design and art using such methodologies. However, due to the different aim of game development, such conclusions should not be conclusive.

REFERENCES

55 per cent revenue share in the new hypercasual developer challenge by CrazyLabs. (2022, August 4). https://www.pocketgamer.biz/55-rev-share-in-the-new-hypercasual-developer-challenge-by-crazylabs/

Adams, E. (2014). *Fundamentals of game design* (Third edition). New Riders.

Ahmad, N. B., Barakji, S. A. R., Shahada, T. M. A., & Anabtawi, Z. A. (2017). How to launch a successful video game: A framework. *Entertainment Computing*, *23*, 1–11. https://doi.org/10.1016/j.entcom.2017.08.001

Aktaş, A. Z., & Orçun, E. (2016). A survey of computer game development. *The Journal of Defense Modeling and Simulation*, *13*(2), 239–251. https://doi.org/10.1177/1548512914554405

Aleem, S., Capretz, L. F., & Ahmed, F. (2016a). A Digital Game Maturity Model (DGMM). *Entertainment Computing*, *17*, 55–73. https://doi.org/10.1016/j.entcom.2016.08.004

Aleem, S., Capretz, L. F., & Ahmed, F. (2016b). Critical Success Factors to Improve the Game Development Process from a Developer's Perspective. *Journal of Computer Science and Technology*, *31*(5), 925–950. https://doi.org/10.1007/s11390-016-1673-z

Almeida, M. S. O., & da Silva, F. S. C. (2013). A Systematic Review of Game Design Methods and Tools. *Entertainment Computing – ICEC 2013* (pp. 17–29). Springer Berlin Heidelberg.

Bates, B. (2004). *Game Design* (2nd ed). Thomson Course Technology.

Bjork, S., & Holopainen, J. (2005). *Patterns in game design* (1st ed). Charles River Media.

Callele, D., Neufeld, E., & Schneider, K. (2005). Requirements engineering and the creative process in the video game industry. *13th IEEE International Conference on Requirements Engineering (RE'05)*, 240–250. https://doi.org/10.1109/RE.2005.58

Capcom. (2024, May 9). *Mobile gaming content market value worldwide from 2011 to 2028 (in billion U.S. dollars) [Graph]*. Statista. https://www.statista.com/statistics/292512/mobile-contents-market-value-worldwide/

Caravella, E., Shivener, R., & Narayanamoorthy, N. (2023). Surveying the Effects of Remote Communication & Collaboration Practices on Game Developers Amid a Pandemic. *Communication Design Quarterly Review*, *10*(4), 5–15. https://doi.org/10.1145/3531210.3531211

Carstensen, P., & Schmidt, K. (2000). *Computer Supported Cooperative Work: New Challenges to Systems Design*.

Chandler, H. M. (2013). *The Game Production Handbook* (3rd Edition). Jones & Bartlett Learning.

Christopoulou, E., & Xinogalos, S. (2017). Overview and Comparative Analysis of Game Engines for Desktop and Mobile Devices. *International Journal of Serious Games*, *4*(4). https://doi.org/10.17083/IJSG.V4I4.194

Chung, S. J., Shim, O., & Kim, S. (2022). Analysing the status of communication designers' role and interdisciplinary collaboration. *Design Journal*, *25*(3), 397–415. https://doi.org/10.1080/14606925.2022.2058682

Church, D. (1999). *Formal Abstract Design Tools*. Gamasutra. http://www.gamasutra.com/view/feature/3357/formal_abstract_design_tools.php

Clarke, V., & Braun, V. (2017). Thematic analysis. *The Journal of Positive Psychology*, *12*(3), 297–298. https://doi.org/10.1080/17439760.2016.1262613

Creswell, J. W. (2009). *Research design: Qualitative, quantitative, and mixed methods approaches* (3rd ed). Sage Publications.

Definition of COLLABORATE. (2024). In *Merriam-Webster.com dictionary*. https://www.merriam-webster.com/dictionary/collaborate

DeWinter, J., & Moeller, R. M. (2016). *Computer Games and Technical Communication: Critical Methods & Applications at the intersection*. Routledge.

Driskell, J. E., Salas, E., & Driskell, T. (2018). Foundations of teamwork and collaboration. *American Psychologist*, *73*(4), 334–348. https://doi.org/10.1037/amp0000241

Ellis, C. A., Barthelmess, P., Chen, J., & Wainer, J. (2005). Person-to-Person Processes: Computer-Supported Collaborative Work. In *Process-Aware Information Systems* (pp. 37–60). John Wiley & Sons, Ltd. https://doi.org/10.1002/0471741442.ch3

Engström, H. (2019, January 1). GDC vs. DiGRA: Gaps in Game Production Research. *Proceedings of DiGRA 2019 Conference: Game, Play and the Emerging Ludo-Mix*. https://dl.digra.org/index.php/dl/article/view/1081

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

Game Developers Conference. (2024). *2024 State of the Game Industry*. https://reg.gdconf.com/state-of-game-industry-2024

Greene, J., & Palmer, L. (2016). It's All Fun and Games Until Someone Pulls Out a Manual: Finding a Role for Technical Communicators in the Game Industry. In J. DeWinter & R. M. Moeller (Eds.), *Computer Games and Technical Communication: Critical Methods & Applications at the intersection*. Routledge.

Grudin, J. (1994a). Computer-supported cooperative work: History and focus. *Computer*, *27*(5), 19–26. https://doi.org/10.1109/2.291294

Grudin, J. (1994b). Groupware and social dynamics: Eight challenges for developers. *Communications of the ACM*, *37*(1), 92–105. https://doi.org/10.1145/175222.175230

Grudin, J., & Poltrock, S. (2014). 27. Computer Supported Cooperative Work. In *Computer Supported Cooperative Work*. https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/computer-supported-cooperative-work

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

Hossain, M. Y., & Zaman, L. (2023). NCCollab: Collaborative behavior tree authoring in game development. *Multimedia Tools and Applications*, *82*(3), 4671–4708. https://doi.org/10.1007/s11042-022-12307-2

Huang, W., Billinghurst, M., Alem, L., Xiao, C., & Rasmussen, T. (2024). *Computer-Supported Collaboration: Theory and Practice* (1st ed.). Wiley. https://doi.org/10.1002/9781119719830

IGDA Education Special Interest Group. (2008). *IGDA Curriculum Framework The Study of Games and Game Development*. www.igda.org

Introducing Whisper. (2024). Retrieved December 5, 2024, from https://openai.com/index/whisper/

Juul, J. (2012). *Casual Revolution: Reinventing Video Games and Their Players*. MIT Press.

Karabinus, A., & Atherton, R. (2018). Games, UX, and the Gaps: Technical Communication Practices in an Amateur Game Design Community. *Proceedings of the 36th ACM International Conference on the Design of Communication*, 1–7. https://doi.org/10.1145/3233756.3233949

Kasurinen, J. (2016). Games as Software: Similarities and Differences between the Implementation Projects. *Proceedings of the 17th International Conference on Computer Systems and Technologies 2016*, 33–40. https://doi.org/10.1145/2983468.2983501

Kasurinen, J., Palacin-Silva, M., & Vanhala, E. (2017). What Concerns Game Developers? A Study on Game Development Processes, Sustainability and Metrics. *2017 IEEE/ACM 8th Workshop on Emerging Trends in Software Metrics (WETSoM)*, 15–21. https://doi.org/10.1109/WETSoM.2017.3

Kasurinen, J., Strandén, J.-P., & Smolander, K. (2013). What do game developers expect from development and design tools? *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*, 36–41. https://doi.org/10.1145/2460999.2461004

Keogh, B. (2015). Between Triple-A, Indie, Casual, and DIY: Sites of tension in the videogames cultural industries. In *The Routledge Companion to the Cultural Industries*. Routledge.

Knezovic, A. (2024, March 12). *Hyper Casual Games: Design, Monetization, and Market Insights*. Udonis Mobile Marketing Agency. https://www.blog.udonis.co/mobile-marketing/mobile-games/hyper-casual-game

Koerber, A., & McMichael, L. (2008). Qualitative Sampling Methods: A Primer for Technical Communicators. *Journal of Business and Technical Communication*, *22*(4), 454–473. https://doi.org/10.1177/1050651908320362

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* (pp. 135–149). Springer. https://doi.org/10.1007/978-3-642-38314-4_10

Kreimeier, B. (2003, March 3). *Gamasutra—Game Design Methods: A 2003 Survey*. Gamasutra. http://www.gamasutra.com/view/feature/2892/game_design_methods_a_2003_survey.php

Le, K. T. (2022, February 23). Region Lock: Turkey's hypercasual truimph, and what it means for its young population. https://www.pocketgamer.biz/region-lock-turkey/

Lee, C. P., & Paine, D. (2015). From The Matrix to a Model of Coordinated Action (MoCA): A Conceptual Framework of and for CSCW. *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 179–194. https://doi.org/10.1145/2675133.2675161

Lehtonen, M., Lu, C., Nummenmaa, T., & Peltonen, J. (2020). Adoption of Requirements Engineering Methods in Game Development: A Literature and Postmortem Analysis. In A. Brooks & E. I. Brooks (Eds.), *Interactivity, Game Creation, Design, Learning, and Innovation* (pp. 436–457). Springer International Publishing. https://doi.org/10.1007/978-3-030-53294-9_32

Liu, J. W., Ho, C. Y., Chang, J. Y. T., & Tsai, J. C. A. (2019). The role of Sprint planning and feedback in game development projects: Implications for game quality. *Journal of Systems and Software*, *154*, 79–91. https://doi.org/10.1016/J.JSS.2019.04.057

Luhova, T., Blazhko, O., Troianovska, Y., & Riashchenko, O. (2019). The canvas-oriented formalization of the game design processes. *2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering, UKRCON 2019 - Proceedings*, 1254–1259. https://doi.org/10.1109/UKRCON.2019.8879899

MagicLab Game Technologies. (2024). https://maglab.com.tr/

Matthews, B., & Ross, L. (2010). *Research Methods: A Practical Guide for the Social Sciences*. Pearson Longman.

Maxim, B. R., & Ridgway, B. (2007). Use of interdisciplinary teams in game development. *2007 37th Annual Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports*, T2H-1-T2H-5. https://doi.org/10.1109/FIE.2007.4417928

Mazzuca, L., Garbugli, A., Sabbioni, A., Bujari, A., & Corradi, A. (2022). Towards a Resource-aware Middleware Support for Distributed Game Engine Design. Proceedings of the 2022 ACM Conference on Information Technology for Social Good, 409–413. https://doi.org/10.1145/3524458.3547126

McDaniel, R. (2015). Communication and knowledge management strategies in video game design and development: A case study highlighting key organizational narratives. *2015 IEEE International Professional Communication Conference (IPCC)*, 1–8. https://doi.org/10.1109/IPCC.2015.7235773

McGrath, J. E. (1991). Time, Interaction, and Performance (TIP): A Theory of Groups. *Small Group Research*, *22*(2), 147–174. https://doi.org/10.1177/1046496491222001

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. *Proceedings - 2021 IEEE/ACM Joint 15th International Conference on Software and System Processes and 16th ACM/IEEE International Conference on Global Software Engineering, ICSSP/ICGSE 2021*, 94–105. https://doi.org/10.1109/ICSSP-ICGSE52873.2021.00019

McKenzie, T., Trujillo, M. M., & Hoermann, S. (2019). Software Engineering Practices and Methods in the Game Development Industry. *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play*

Companion Extended Abstracts, 181–193. https://doi.org/10.1145/3341215.3354647

Merriam, S. B. (2014). *Qualitative Research: A Guide to Design and Implementation* (3. Aufl). Jossey-Bass.

Moreno-Ger, P., Sierra, J. L., Martínez-Ortiz, I., & Fernández-Manjón, B. (2007). A documental approach to adventure game development. *Science of Computer Programming*, *67*(1), 3–31. https://doi.org/10.1016/j.scico.2006.07.003

Neil, K. (2012). Game design tools: Time to evaluate. *Proceedings of Nordic DiGRA 2012 Conference*. https://dl.digra.org/index.php/dl/article/view/606

Nissani, M. (1995). Fruits, Salads, and Smoothies: A Working Definition of Interdisciplinarity. *The Journal of Educational Thought (JET) / Revue de La Pensée Éducative*, *29*(2), 121–128.

O Connor, M., Conboy, K., & Dennehy, D. (2022). COVID-19 affected remote workers: A temporal analysis of information system development during the pandemic. *Journal of Decision Systems*, *31*(3), 207–233. https://doi.org/10.1080/12460125.2020.1861772

Oliveira, A., Lopes, I., & Costa, C. (2015). Tools for online collaboration: Do they contribute to improve teamwork? *Tools for Online Collaboration: Do They Contribute to Improve Teamwork?*, 105–112. https://doi.org/10.5901/mjss.2015.v6n6s4p511

Olson, G. M., & Olson, J. S. (2012). Collaboration Technologies. In J. A. Jacko, *Human Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications, Third Edition*. Taylor & Francis Group.

One Page Design—GDC Presentations. (2010). https://stonetronix.com/gdc-2010/

Patel, H., Pettitt, M., & Wilson, J. R. (2012). Factors of collaborative working: A framework for a collaboration model. *Applied Ergonomics*, *43*(1), 1–26. https://doi.org/10.1016/j.apergo.2011.04.009

Patterns—Gdp3. (n.d.). Retrieved November 24, 2024, from http://virt10.itu.chalmers.se/index.php/Category:Patterns

Pizzo, A. D. (2023). Hypercasual and Hybrid-Casual Video Gaming: A Digital Leisure Perspective. *Leisure Sciences*, 1–20. https://doi.org/10.1080/01490400.2023.2211056

Politowski, C., Petrillo, F., Ullmann, G. C., de Andrade Werly, J., & Guéhéneuc, Y.-G. (2020). Dataset of Video Game Development Problems. *Proceedings of the 17th International Conference on Mining Software Repositories*, 553–557. https://doi.org/10.1145/3379597.3387486

Politowski, C., Petrillo, F., Ullmann, G. C., & Guéhéneuc, Y.-G. (2021). Game industry problems: An extensive analysis of the gray literature. *Information and Software Technology*, 134, 106538. https://doi.org/10.1016/j.infsof.2021.106538

Principles behind the Agile Manifesto. (2001). Retrieved November 25, 2024, from https://agilemanifesto.org/principles.html

Rabionet, S. E. (2011). How I Learned to Design and Conduct Semi-Structured Interviews: An Ongoing and Continuous Journey. *Qualitative Report*, *16*(2), 563–566.

Salazar, M. G., Mitre, H. A., Olalde, C. L., & Sánchez, J. L. G. (2012). Proposal of game design document from software engineering requirements perspective. *2012 17th International Conference on Computer Games (CGAMES)*, 81–85. https://doi.org/10.1109/CGames.2012.6314556

Saldana, J. (2011). *Fundamentals of Qualitative Research*. Oxford University Press.

Sanders, E., & Stappers, P. (2012). *Convivial toolbox: Generative research for the front end of design*. Bis.

Schmidt, K., & Bannon, L. (1992). Taking CSCW seriously. *Computer Supported Cooperative Work (CSCW)*, *1*(1), 7–40. https://doi.org/10.1007/BF00752449

Schwaber, K., & Sutherland, J. (n.d.). *Scrum Guide | Scrum Guides*. Retrieved November 25, 2024, from https://scrumguides.org/scrum-guide.html

Stardew Valley—FAQ. (2024). Retrieved November 29, 2024, from https://www.stardewvalley.net/faq/

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Harvard Business Review*, *64*(1), 137–146.

Tanner, M., & Naidoo, M. (2021, August 9). The Use of Tools Affordances to Support Communication and Collaboration during COVID-19 remote work. *AMCIS 2021 Proceedings*. https://aisel.aisnet.org/amcis2021/virtual_communities/virtual_communities/8

The Hypercasual business model. (2021, July 7). Homa. https://homa.vercel.app/

Timmermann, S., Staegemann, D., Volk, M., Pohl, M., Haertel, C., Hintsch, J., & Turowski, K. (2022). Facilitating the Decentralisation of Software Development Projects from a Project Management Perspective: A Literature Review: *Proceedings of the 4th International Conference on Finance, Economics, Management and IT Business*, 22–34. https://doi.org/10.5220/0010938200003206

Tran, M. Q., & Biddle, R. (2009). An ethnographic study of collaboration in a game development team. *Loading...*, *3*(5), Article 5.

Ullmann, G. C., Politowski, C., Guéhéneuc, Y.-G., Petrillo, F., & Montandon, J. E. (2022). Video game project management anti-patterns. *Proceedings of the 6th International ICSE Workshop on Games and Software Engineering:*

Engineering Fun, Inspiration, and Motivation, 9–15. https://doi.org/10.1145/3524494.3527623

Whitson, J. R. (2018). Voodoo software and boundary objects in game development: How developers collaborate and conflict with game engines and art tools. *New Media & Society*, *20*(7), 2315–2332. https://doi.org/10.1177/1461444817715020

Yang, L., Holtz, D., Jaffe, S., Suri, S., Sinha, S., Weston, J., Joyce, C., Shah, N., Sherman, K., Hecht, B., & Teevan, J. (2022). The effects of remote work on collaboration among information workers. *Nature Human Behaviour*, *6*(1), 43–54. https://doi.org/10.1038/s41562-021-01196-4

Zagal, J. P., Mateas, M., Fernández-Vara, C., Hochhalter, B., & Lichti, N. (2005). Towards an ontological language for game analysis. *Proceedings of DiGRA 2005 Conference: Changing Views: Worlds in Play*.

APPENDICES

A. Consent Form - Interviews

ARAŞTIRMAYA GÖNÜLLÜ KATILIM FORMU

Bu araştırma, ODTÜ Endüstriyel Tasarım Bölümü Yüksek Lisans öğrencisi İzel Çelik tarafından Dr. Öğretim Üyesi Güzin Şen danışmanlığında yürütülen "Mobil oyun geliştirme sırasında birlikte çalışmayı geliştirme: tasarım müdahale alanları" başlıklı yüksek lisans tezi kapsamında yürütülmektedir. Bu form sizi araştırma koşulları hakkında bilgilendirmek için hazırlanmıştır.

Çalışmanın Amacı Nedir?

Araştırmanın amacı, mobil oyun geliştirme sırasında ekip üyeleri arasındaki tasarım iletişimini incelemek ve ekipteki iletişim ve iş birliğinin geliştirilmesine yönelik tasarım müdahalesi alanlarını bulmaktır. Bunun için mülakat, gözlem ve odak grubu çalışmaları yapılacaktır.

Bize Nasıl Yardımcı Olmanızı İsteyeceğiz?

Araştırmaya katılmayı kabul ederseniz, sizden yaklaşık bir saat sürmesi beklenen bir mülakat çalışmasına katılmanız beklenmektedir. Bu mülakatta oyun geliştirme sürecindeki iş birliği hakkında açık uçlu sorular sorulacaktır ve içerik analizi ile değerlendirilmek üzere cevaplarınızın ses kaydı alınacaktır. Mülakat sırasında örnek göstereceğiniz araçların fotoğrafları gizlilik ilkelerini gözetecek şekilde tezde anonimleştirilerek kullanılmak üzere kaydedilecektir.

Sizden Topladığımız Bilgileri Nasıl Kullanacağız?

Araştırmaya katılımınız tamamen gönüllülük temelinde olmalıdır. Çalışmada sizden kimlik veya kurum belirleyici hiçbir bilgi istenmemektedir. Cevaplarınız tamamıyla gizli tutulacak ve sadece araştırmacılar tarafından değerlendirilecektir. Katılımcılardan elde edilecek tüm veriler toplu halde değerlendirilecek ve anonimleştirilerek bilimsel yayımlarda kullanılacaktır.

Katılımınızla ilgili bilmeniz gerekenler:

Çalışma, genel olarak kişisel rahatsızlık verecek sorular veya uygulamalar içermemektedir. Ancak, katılım sırasında sorulardan ya da herhangi başka bir nedenden ötürü kendinizi rahatsız hissederseniz çalışmayı yarıda bırakıp çıkmakta serbestsiniz. Böyle bir durumda çalışmayı uygulayan kişiye çalışmadan çıkmak istediğinizi söylemek yeterli olacaktır.

Araştırmayla ilgili daha fazla bilgi almak isterseniz:

Bu çalışmaya katıldığınız için şimdiden teşekkür ederiz. Çalışma hakkında daha fazla bilgi almak için ODTÜ Endüstriyel Tasarım Bölümü öğretim üyelerinden Dr. Öğretim Üyesi Güzin Şen (E-posta: [redacted]) ya da yüksek lisans öğrencisi İzel Çelik (E-posta: [redacted]) ile iletişim kurabilirsiniz.

Yukarıdaki bilgileri okudum ve bu çalışmaya tamamen gönüllü olarak katılıyorum.

(Formu doldurup imzaladıktan sonra uygulayıcıya geri veriniz).

İsim Soyisim Tarih İmza

---/----/-----

B. Interview Questions (Turkish)

Merhaba, tez çalışmama katıldığınız için teşekkür ederim. Ben İzel, bugün sizinle oyun yapım sürecindeki deneyimleriniz hakkında bir mülakat yapacağız.
Çalışmanın amacı mobil oyun geliştirme sırasında ekip üyeleri arasındaki tasarım iletişimini incelemek ve tasarım müdahalesi alanlarını bulmak.
Önceden de belirttiğim gibi, cevaplarınız tamamıyla gizli tutulacak ve sadece araştırmacılar tarafından değerlendirilecek. Siz ve diğer katılımcılardan elde edilen bilgiler topluca değerlendirilecek ve sadece bilimsel yayınlarda ve raporlarda kullanılacak. Sağladığınız veriler, bu yayın ve raporlarda, kimlik bilgileriniz ile kesinlikle eşleştirilmeyecek.
Araştırmanın herhangi bir aşamasında kendinizi rahatsız hissederseniz katılımı yarıda bırakıp çıkmakta serbestsiniz. Böyle bir durumda çalışmadan çıkmak istediğinizi söylemeniz yeterli.

Oyun Geliştirme Süreci

Soruların ilk kısmı oyun geliştirme üzerine.

1. Ekibiniz hangi türde oyun geliştiriyor?
2. Ekibinizin oyun geliştirme sürecinizi kısaca anlatabilir misiniz? Bitirdiğiniz bir oyun projesi üzerinden örneklendirerek de anlatabilirsiniz.
3. Oyun geliştirken hangi araçları kullanıyorsunuz? Örnek olarak oyun motorları, tasarım araçları, IDE'ler (tümleşik geliştirme ortamı), iletişim araçları (Slack, Discord gibi) ve görev yöneticiler (Clickup, Jira gibi) olabilir.
4. Ekip içindeki iş birliğini düşündüğünüzde, bu araçlardan hangileri yaptığınız işte en çok etkili?

Oyun Yapımındaki Roller ve İş birliği

Soruların ikinci kısmı ise ekip rolleri ve roller arasındaki iş birliği ve iletişim hakkında.

5. Ekibinizdeki rolünüz nedir?
    a. Bir oyun projesindeki sorumluluklarınız neler?
    b. Ekipteki diğer rolleri kısaca tanımlayabilir misiniz?
6. Ne sıklıkla tüm ekibi içeren toplantılar yapıyorsunuz?
    c. Tüm ekibi içeren toplantılar dışında sıklıkla katıldığınız toplantılar genellikle hangi rolleri içeriyor?
7. Toplantılar dışında ekip içindeki iletişiminizi nasıl sağlıyorsunuz?
8. Oyun geliştirme sürecinde ekip içinde oyun hakkında neleri aktarmanız gerekiyor? Örnek vererek anlatabilir misiniz?
    a. Yazılı veya sözlü olarak ilettiğiniz içerikler neler?
    b. Görsel kullanarak ilettiğiniz içerikler neler?
    c. Bunlar dışında bir yöntemle iletmeniz gereken içerikler neler?
9. Oyundaki farklı alanları düşünürsek, hikaye, bölüm tasarımı, kodlama gibi; farklı alanların iletişiminde farklar oluyor mu? Oluyorsa nasıl farklılıklar oluyor? Örnek verebilir misiniz?
10. Oyun tasarım kararlarını iletirken karşılaştığınız en büyük zorluk nedir?

11. Oyun tasarım kararlarını iletmek için nasıl sistemler kullanıyorsunuz?
12. Kullandığınız herhangi bir araç var mı? Bu araçları göstermeniz mümkün mü?
    a. Kullandığınız araçların kullandığınız iş için tasarlandığını düşünüyor musunuz?
    b. Hangi kısımlar bu iş için tasarlandığını düşündürüyor? Hangi kısımların kullandığınız işe uygun olmadığını düşünüyorsunuz?
    c. Bu amaçla oluşturduğunuz geçici çözümler var mı? Bu çözümün ilgili en memnun olduğunuz kısmı nedir?
    d. Kullandığınız sistemin sorun olduğunu düşündüğünüz yönleri var mı? Bu yönleri neler sorun olarak tanımlamanıza neden oluyor? Bu sorunları aşmak için oluşturduğunuz geçici çözümler var mı?
13. Daha önce bu amaçla denediğiniz ancak şu anda kullanmadığınız herhangi bir araç var mı?
    a. Bu aracı şu an kullanmıyor olmanızı neye bağlıyorsunuz?
14. Oyun geliştirme sürecindeki oyun hakkındaki iletişimlerde sizin için en önemli nokta nedir?
15. Ekip içinde geri bildirimleri nasıl yönetiyorsunuz?
    a. Geri bildirimlerinizi hangi yöntemlerle iletiyorsunuz?
    b. Geçmişte verilmiş geri bildirimlere dönmeniz gerekiyor mu? Gerekiyorsa hangi amaçlarla gerekiyor? Gerekiyorsa bunu nasıl yapıyorsunuz?
16. Az önceki sorulara verdiğiniz cevapları da göz önünde bulundurarak, oyun tasarım kararlarının iletişimini geliştirecek ne olurdu?
17. Oyun tasarım kararlarının iletişimi için bir araç tasarlanacak olsaydı, hangi özelliklerin kesinlikle olmasını isterdiniz?
18. Bu tür bir araçta dikkat edilmesi gerektiğini düşündüğünüz noktalar neler?

Demografi

Son olarak da demografik sorularımız var.

19. Hangi şehirde yaşıyorsunuz? (Farklı şehirde çalışılıyor ise) Hangi şehirde çalışıyorsunuz?
20. Hangi yaş aralığındasınız? 18-25, 26-35, 36-45, 46-55, 56-65, 65 ve üzeri.
21. Eğitim seviyeniz nedir?
22. Çalışma sisteminizi hangisi daha iyi karşılar? Uzaktan, hibrit, yüz yüze.
23. Güncel iş ünvanınız nedir?
24. Ne kadar süredir içinde bulunduğunuz ekiple çalışıyorsunuz?
25. Size sormadığım ancak bu çalışma için değerli olacağını düşündüğünüz herhangi bir şey var mı?

Sorularımız bu kadar. Eklemek istediğiniz bir şey var mı?

Görüşmeyi burada sonlandırabiliriz, eğer çalışmamız ve sonuçları ile ilgili daha fazla bilgi almak isterseniz, çalışma sona erdiğinde size iletişim bilgileriniz aracılığıyla bilgilendirme sağlayabiliriz. Değerli vaktinizi ayırdığınız için çok teşekkür ederiz. İyi günler dilerim.

C. Interview Questions (English)

Greetings, thank you for participating in my thesis study. I am İzel, and today I will interview you about your experiences in game development processes. The purpose of this study is to observe the design communication between team members and find areas for design intervention.
As I mentioned earlier, your answers will be kept private, and they will only be evaluated by the researchers. The information obtained from you and the other participants will be evaluated collectively and they will only be used for scientific publications and reports. The information you provide will not be matched with your identity within these reports and publications.
If at any point during the research you feel uncomfortable, you are free to terminate the study and leave. In such an occasion you only need to declare that you want to leave.

Game Development Process

The first part of the questions is about game development.

1. What kind of games do your team develop?
2. Can you briefly describe your team's game development process? You can use examples from a game project that you completed.
3. What tools do you use during game development? Examples may include: game engines, design tools, IDEs (Integrated Development Environments), communication tools (Slack, Discord etc.) and task managers (Clickup, Jira etc.).
4. Considering collaboration between team members, which of these tools has a greater effect on the work you do?

Roles and Collaboration in Game Development

The second part of the questions is related to team roles, collaboration and communication.

5. What is your role in your team?
    a. What are your responsibilities in a game project?
    b. Can you briefly define other roles in your team?
6. How often do you do meetings with all of the team members?
    a. What roles usually include in the meetings that you are present and that do not include all the team members?
7. How do you communicate with your team in addition to the meetings?
8. What do you need to convey regarding the game during game development? Can you give examples?
    a. What contents do you share with your team verbally or through text?
    b. What contents do you share using visuals?
    c. What contents do you share using other methods?
9. Considering different areas in game like story, level design, or coding, are there differences in communication for different areas? If so, what are the differences? Can you share examples?

10. What is the largest challenge you face while conveying game design decisions?
11. What are the systems you use to convey game design decisions?
12. Are there any tools you use? Is it possible for you to show these tools?
    a. Do you think the tools you use are designed for your work's purposes?
    b. Which parts make you think it was designed for this work? Which parts do you think are not suitable for your work?
    c. Are there any temporary solutions you developed for this purpose? Which part of this solution are you most satisfied with?
    d. Are there any aspects of your system that you think cause problems? What makes you define these aspects as problems? Do you have any temporary solutions to overcome these problems?
13. Are there any tools you tried before but are not using currently?
    a. Why are you not using this tool currently?
14. What is the most important point in game development communication for you?
15. How do you manage feedback within your team?
    a. What are the methods you use to convey your feedbacks?
    b. Do you need to return to previous feedbacks? If so, for what purpose? If so, how do you do this?
16. Considering your answer to earlier questions, what would improve the communication of game design decisions?
17. If you were designing a tool for communicating design decisions, what properties would you definitely want to include?
18. What are the points you think should be considered for such a tool?

Demography

Finally, we have demographic questions

19. What city do you live in? (If they are working in a different city, what city do you work in?)
20. What age group are you in? 18-25, 26-35, 36-45, 46-55, 56-65, 65+.
21. What is your level of education?
22. Which better defines your work system? Remote, hybrid, on-site.
23. What is your current job title?
24. How long have you been working with the team you are currently in?
25. Is there anything I have not asked you that you think would be valuable for this study?

That's the end of our questions. Is there anything you'd like to add?

We can terminate our session now. If you want to learn more about our study and our results, we can inform you using the information you provided. Thank you for sharing your valuable time. Have a nice day.

D. Consent Form – Generative Workshop (Focus Group)

ARAŞTIRMAYA GÖNÜLLÜ KATILIM FORMU

Bu araştırma, ODTÜ Endüstriyel Tasarım Bölümü Yüksek Lisans öğrencisi İzel Çelik tarafından Dr. Öğretim Üyesi Güzin Şen danışmanlığında yürütülen "Mobil oyun geliştirme sırasında birlikte çalışmayı geliştirme: tasarım müdahale alanları" başlıklı yüksek lisans tezi kapsamında yürütülmektedir. Bu form sizi araştırma koşulları hakkında bilgilendirmek için hazırlanmıştır.

Çalışmanın Amacı Nedir?

Araştırmanın amacı, mobil oyun geliştirme sırasında ekip üyeleri arasındaki tasarım iletişimini incelemek ve ekipteki iletişim ve iş birliğinin geliştirilmesine yönelik tasarım müdahalesi alanlarını bulmaktır. Bunun için mülakat, gözlem ve odak grubu çalışmaları yapılacaktır.

Bize Nasıl Yardımcı Olmanızı İsteyeceğiz?

Araştırmaya katılmayı kabul ederseniz, sizden yaklaşık iki saat süren yaratıcı tasarım araştırması yöntemlerini içeren bir odak grubu çalışmasına katılmanız beklenmektedir. Oturumlarda ses ve görüntü kaydı alınacak ve katılımcıların üzerinde çalıştığı materyaller fotoğraflanacaktır. Yapılan tüm kayıtlar anonimleştirilerek incelenecektir.

Sizden Topladığımız Bilgileri Nasıl Kullanacağız?

Araştırmaya katılımınız tamamen gönüllülük temelinde olmalıdır. Çalışmada sizden kimlik veya kurum belirleyici hiçbir bilgi istenmemektedir. Konuşmalarınız tamamıyla gizli tutulacak ve sadece araştırmacılar tarafından değerlendirilecektir. Katılımcılardan elde edilecek tüm veriler toplu halde değerlendirilecek ve anonimleştirilerek bilimsel yayımlarda kullanılacaktır.

Katılımınızla ilgili bilmeniz gerekenler:

Çalışma, genel olarak kişisel rahatsızlık verecek sorular veya uygulamalar içermemektedir. Ancak, katılım sırasında sorulardan ya da herhangi başka bir nedenden ötürü kendinizi rahatsız hissederseniz çalışmayı yarıda bırakıp çıkmakta serbestsiniz. Böyle bir durumda çalışmayı uygulayan kişiye çalışmadan çıkmak istediğinizi söylemek yeterli olacaktır.

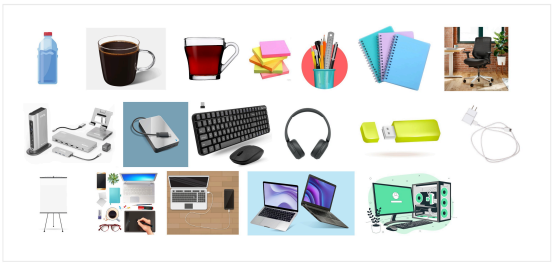
Araştırmayla ilgili daha fazla bilgi almak isterseniz:

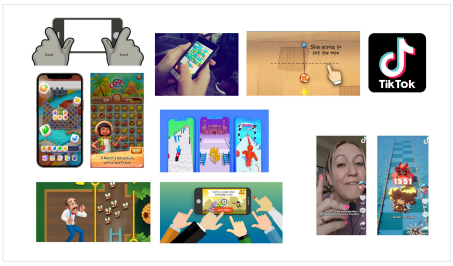
Bu çalışmaya katıldığınız için şimdiden teşekkür ederiz. Çalışma hakkında daha fazla bilgi almak için ODTÜ Endüstriyel Tasarım Bölümü öğretim üyelerinden Dr. Öğretim Üyesi Güzin Şen (E-posta: [redacted]) ya da yüksek lisans öğrencisi İzel Çelik (E-posta: [redacted]) ile iletişim kurabilirsiniz.

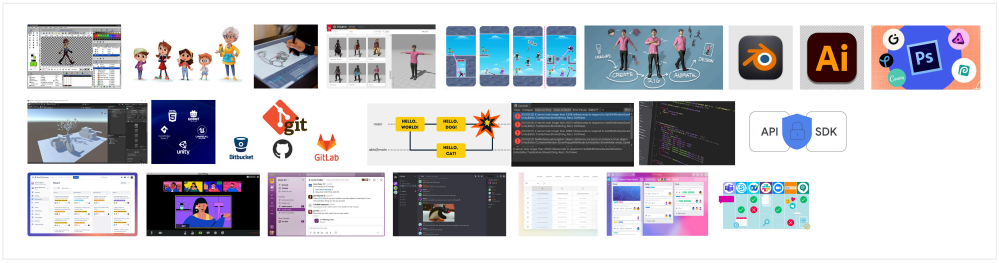
Yukarıdaki bilgileri okudum ve bu çalışmaya tamamen gönüllü olarak katılıyorum.
(Formu doldurup imzaladıktan sonra uygulayıcıya geri veriniz).

İsim Soyisim	Tarih	İmza
	---/----/-----	

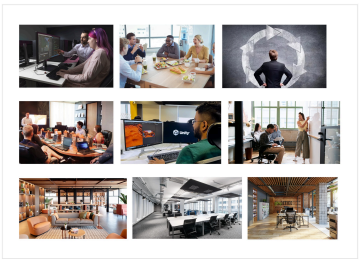
E. Toolbox used in Generative Workshop



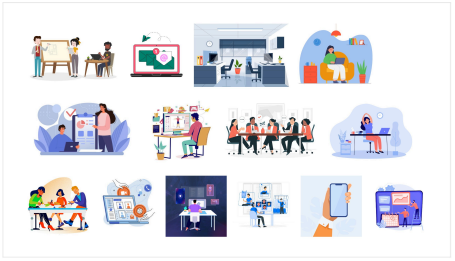


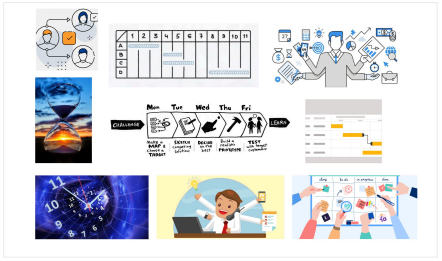




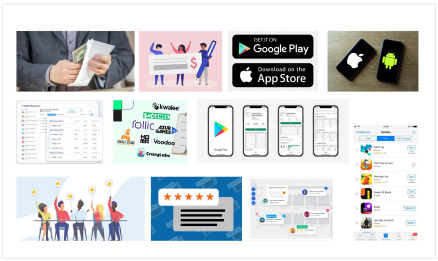








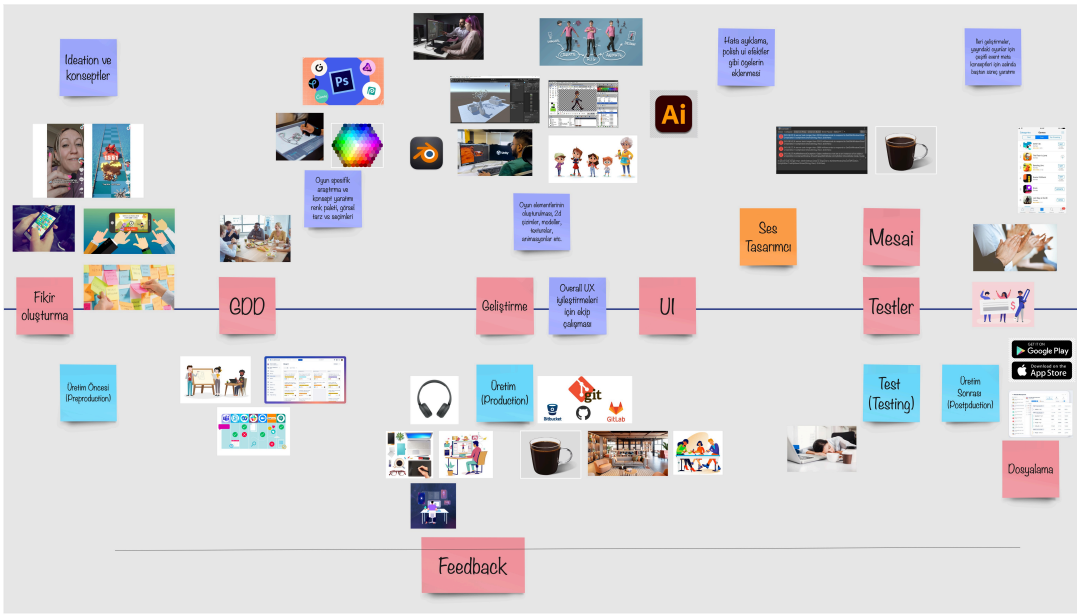


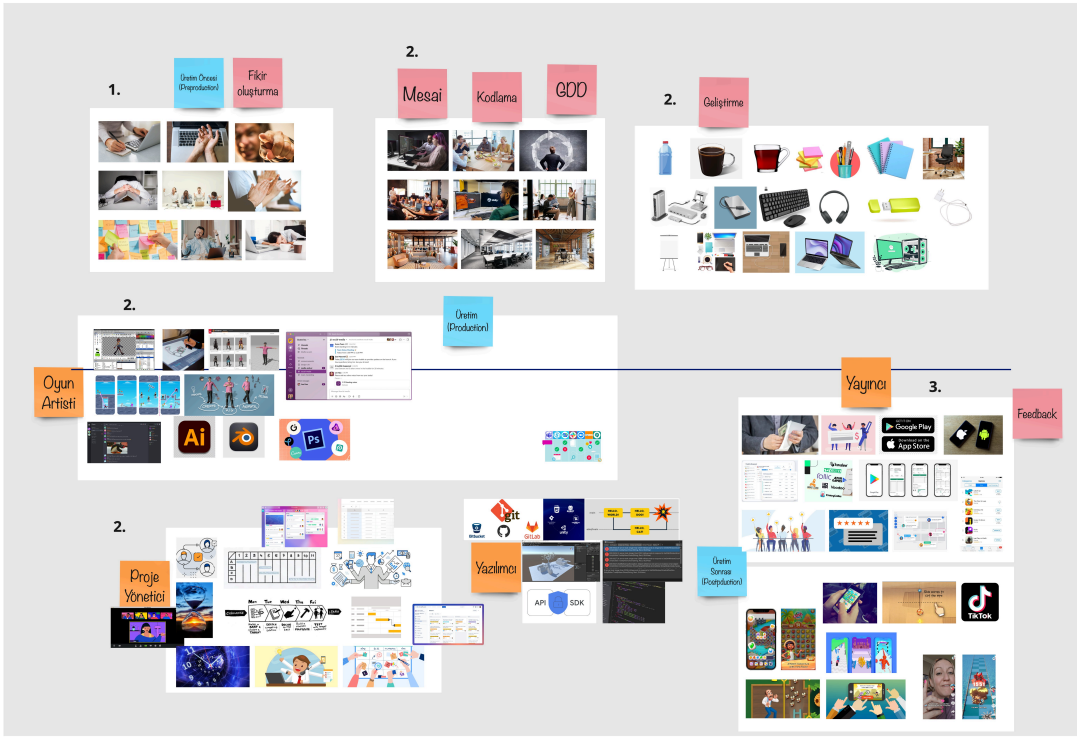


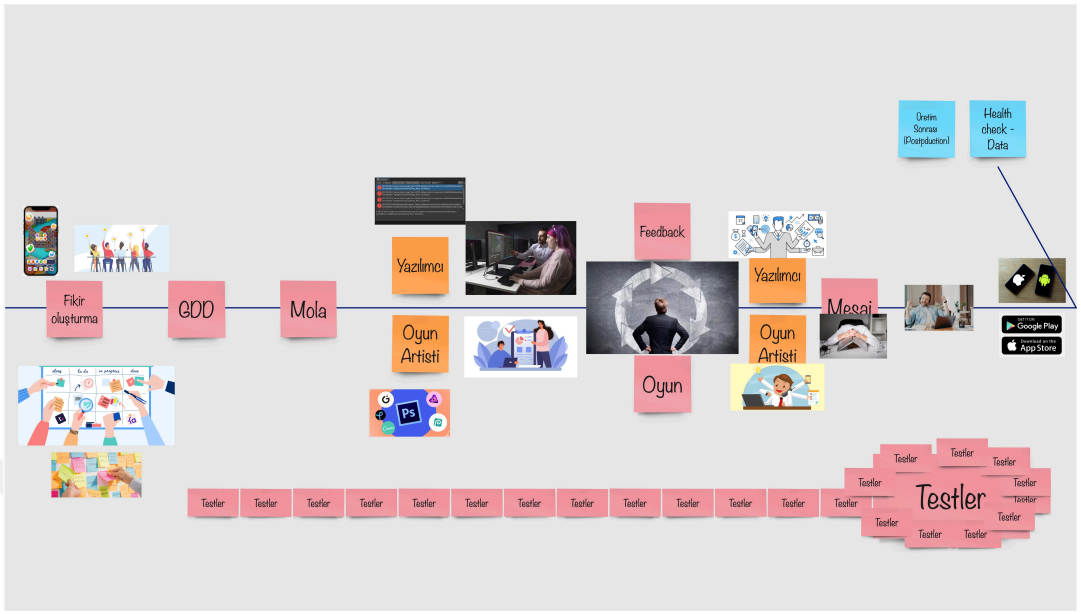


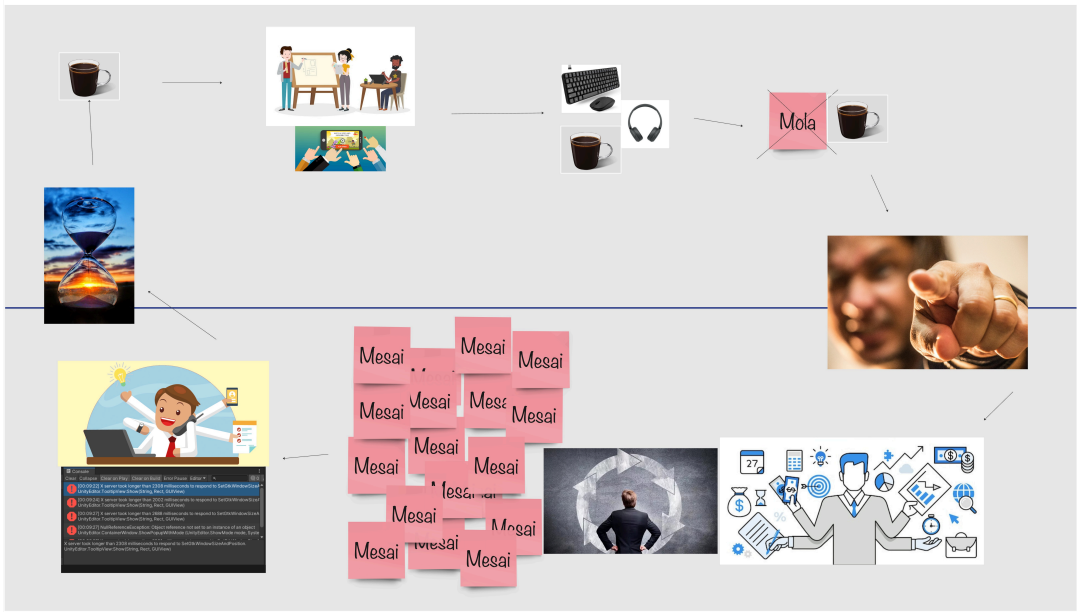
F. Generative Workshop Outcomes

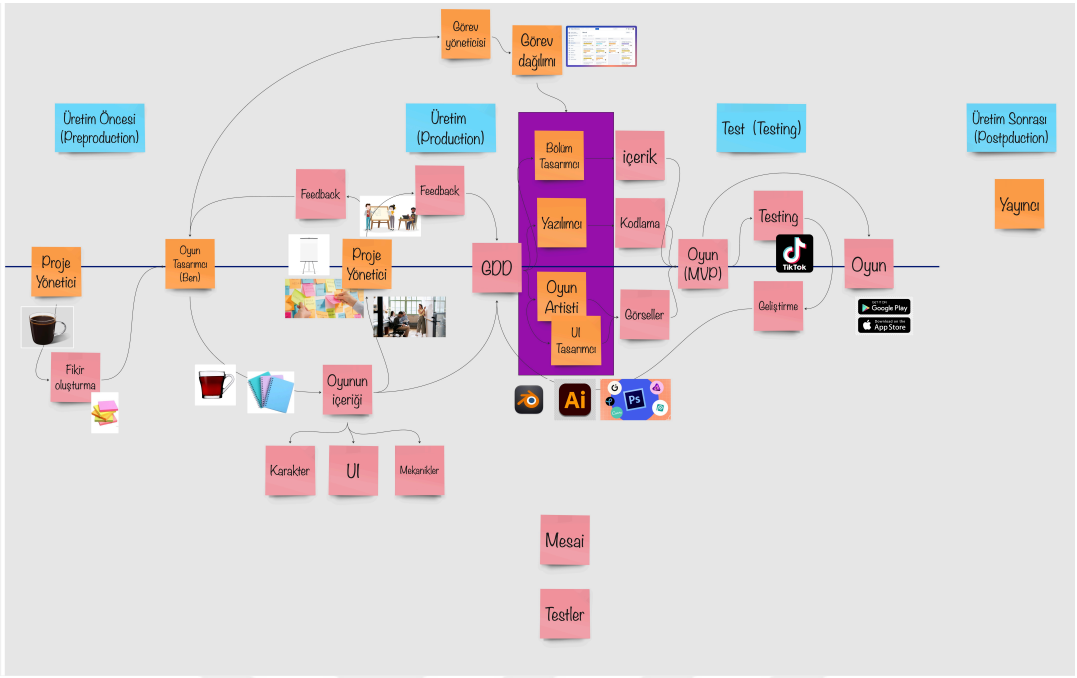
Collages Created by Participants





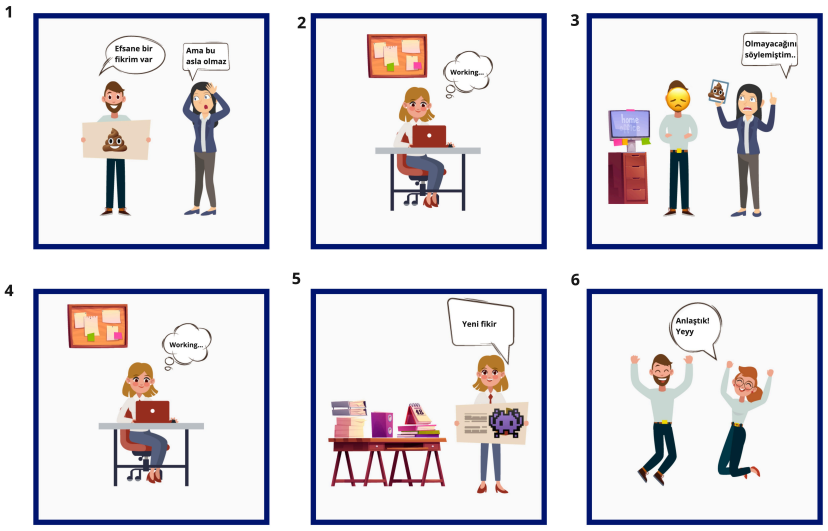

Üretim Sonrası (Postproduction)
Health check - Data
Fikir oluşturma
GDD
Mola
Yazılımcı
Oyun Artisti
Feedback
Oyun
Yazılımcı
Oyun Artisti
Mesai
Testler


Mola
Mesai

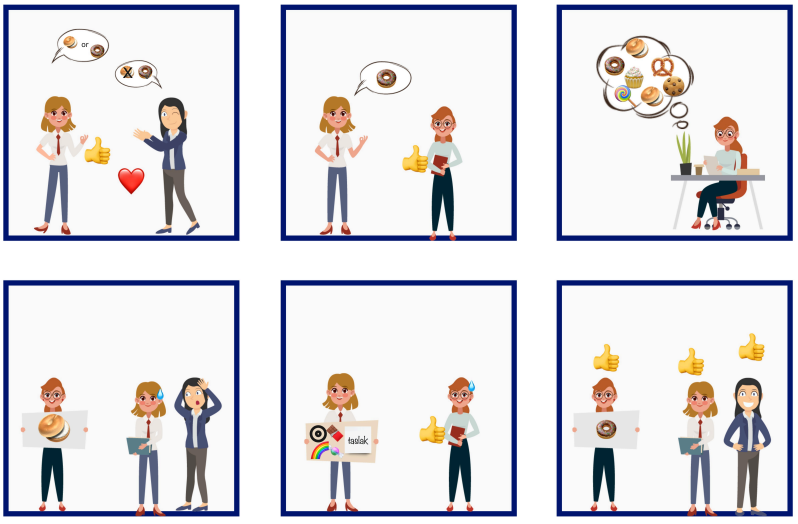

Görev yöneticisi
Görev dağılımı
Üretim Öncesi (Preproduction)
Üretim (Production)
Test (Testing)
Üretim Sonrası (Postpduction)
Bölüm Tasarımcı
içerik
Feedback
Feedback
Yazılımcı
Kodlama
Testing
Yayıncı
Proje Yönetici
Oyun Tasarımcı (Ben)
Proje Yönetici
GDD
Oyun (MVP)
Oyun
Oyun Artisti
Görseller
Geliştirme
UI Tasarımcı
Fikir oluşturma
Oyunun içeriği
Karakter
UI
Mekanikler
Mesai
Testler

Comic Strips Created by Participants

Pair 1



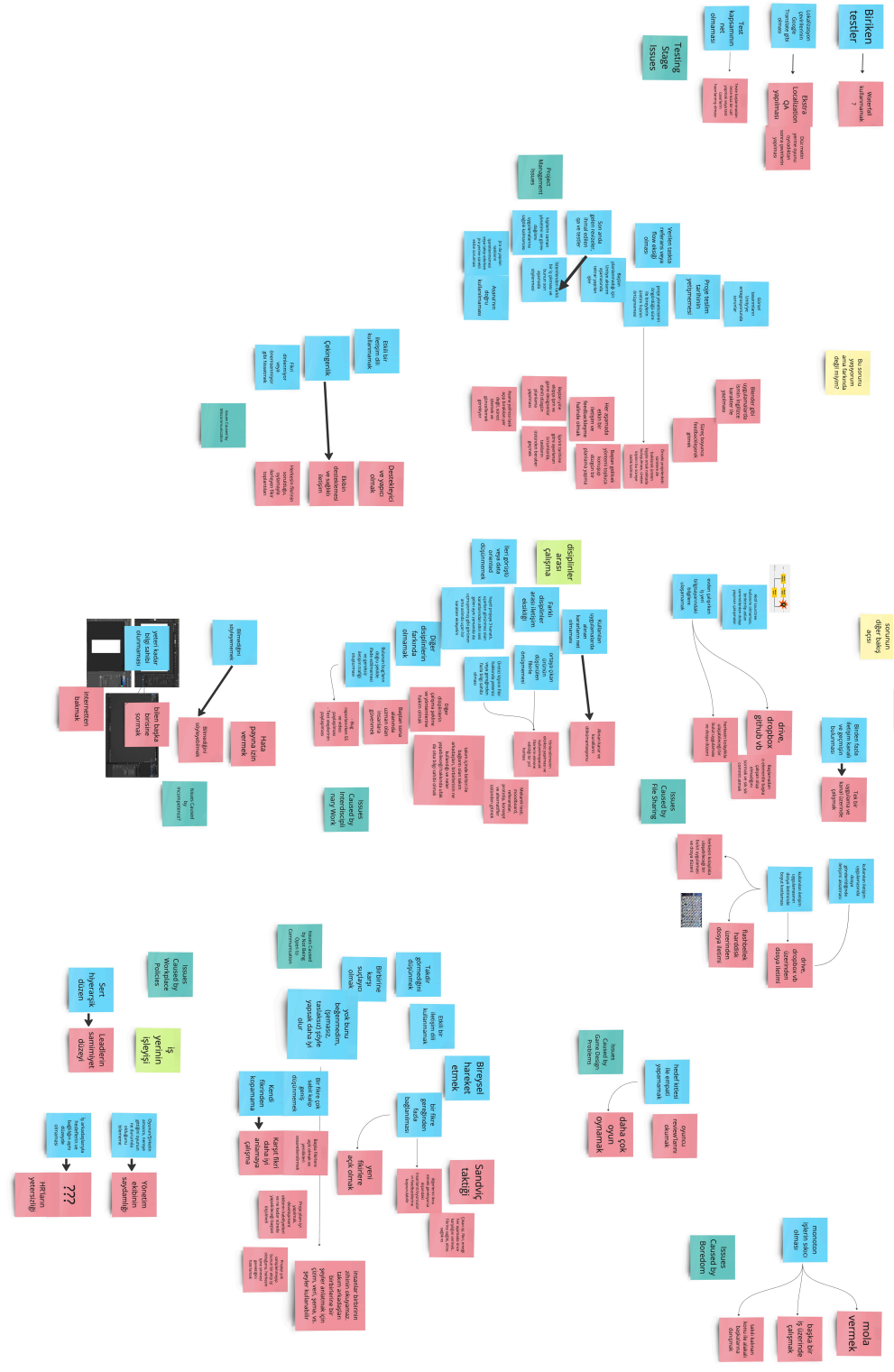
Pair 3



Pair 2



Brainstorming Outcomes


Birliken testler
Testing Stage Issues
Project Management Issues
Issues Caused by Boredom
Issues Caused by Workplace Policies
Issues Caused by Game Design Problems
Issues Caused by Interdisciplinary Work
disiplinler arası çalışma
dropbox
github vb.
daha çok oyun oynamak
Sandviç taktiği
Birleşsel hareket etmek
mola vermek
başka bir iş üzerinde çalışmak
Yönetim
İş yerinin işleyişi
Sert hiyerarşik düzen
Hata payına izin vermek
Bilmediğin soru yanıtlanmaz
Bilmediğini söyleyebilmek
Yeterli kadar bilgi olmaması
Bilen başka birini sormak
İnternetten bakmak
Çokeşgenlik
Destekleyici ve yapıcı olmak
Ekibin ilerlemesini engellemek
Ekibin ilerlemesini sağlamak
Ekibin kapsamını kullanmamak
Fikri dayatmak yerine sorgulamak
Herkesin fikirlerini dinlemek
Her aşamada ihtiyacın ne olduğunu bilmek
Hedefleri yerine getirmek
Hedefleri yerine getirmemek
Hedefleri yerine getirmek
Hedefleri yerine getirmemek
Hedefleri yerine getirmek
Hedefleri yerine getirmem