Revista Brasileira de Computação Aplicada, Novembro, 2025

DOI: 10.5335/rbca.v17i3.16407
Vol. 17, Nº 3, pp. 74–83
Homepage: seer.upf.br/index.php/rbca/index

ORIGINAL PAPER

Modelos de desenvolvimento de software para desenvolvimento de jogos: uma revisão de literatura

Software Development Models in Game Development: a Literature Review

Gabriel Acassio Correia [1], Vinícius Carvalho Pereira [2], Cristiano Maciel [3]

[1,3]Instituto de Computação/Universidade Federal de Mato Grosso - IC/UFMT, [2]Instituto de Linguagens/Universidade Federal de Mato Grosso - IL/UFMT

gabriel.acassio.correia@gmail.com; †vinicius.pereira@ufmt.br;‡cristiano.maciel@ufmt.br

Recebido: 19/10/2024. **Revisado:** 28/10/2025. **Aceito:** 14/11/2025.

Resumo

O presente trabalho é uma revisão sistemática de literatura (RSL) baseada no framework PICOC (Population, Intervention, Comparison, Outcome, and Context), com o objetivo de mapear e analisar estudos sobre a aplicação de metodologias de engenharia de software para o desenvolvimento de jogos digitais. Foram analisados artigos científicos até 2024 recuperáveis nas bases ACM, IEEE e SOL por meio da string de busca ("game development model"OR "game development models") OR (("software development model"OR "software development models") AND (games OR game)). Como resultado, foram encontrados 9 artigos que atendiam aos critérios de inclusão, sobre os quais apresentamos as principais metodologias aplicadas, suas características, contexto de aplicação e desempenho. De modo geral, a análise dos artigos selecionados revela que, embora existam paralelos significativos com o desenvolvimento de software tradicional, o desenvolvimento de jogos digitais apresenta desafios e demandas únicas que requerem metodologias adaptadas e específicas. Essa distinção se torna evidente na maneira como os modelos de engenharia de software são aplicados e modificados para atender às necessidades específicas do setor de jogos, que combina de forma intrínseca aspectos técnicos e criativos.

Palavras-Chave: Desenvolvimento de jogos digitais; modelos de desenvolvimento de software; revisão sistemática de literatura.

Abstract

The present work is a systematic literature review (SLR) based on the PICOC framework (Population, Intervention, Comparison, Outcome, and Context), with the objective of mapping and analyzing studies on the application of software engineering methodologies to game development. We analyzed scientific papers up to 2024, retrievable from the ACM, IEEE, and SOL databases using the search string ("game development model"OR "game development models") OR (("software development model"OR "software development models") AND (games OR game)). As a result, 9 articles that met the inclusion criteria were found, and we present the main methodologies applied, their characteristics, application context, and performance. Overall, the analysis of the selected articles reveals that, although there are significant parallels with traditional software development, game development presents unique challenges and demands that require adapted and specific methodologies. This distinction becomes evident in how software engineering models are applied and modified to meet the specific needs of the gaming industry, which intrinsically combines technical and creative aspects.

Keywords: Digital games development; software development models; systematic literature review.

1 Introdução

Os jogos digitais desempenham um papel social significativo na contemporaneidade, atuando não apenas como formas de entretenimento, mas também como manifestações artísticas interativas que podem transmitir narrativas complexas e evocar profundas emoções (Marcos and Zagalo, 2011). Para tanto, a diversidade temática presente nos jogos é vasta, abrangendo desde épicas aventuras de fantasia até reflexões filosóficas sobre a condição humana. Essa variedade temática se reflete na rica produção artística dos jogos, que combina visuais deslumbrantes, trilhas sonoras imersivas e mecânicas de jogo inovadoras, resultando em experiências interativas únicas e envolventes.

A indústria de jogos é uma das mais dinâmicas e lucrativas dentro do setor de entretenimento e de produção de software, com receitas globais que superam centenas de bilhões de dólares anualmente (Read, 2022). Estados Unidos e China são as nações que abrigam as maiores corporações da indústria de jogos, detendo títulos de grande orçamento que agregam milhões de jogadores em escala mundial. Porém, para além das corporações, o desenvolvimento independente (indie) também ocupa um espaço crucial, permitindo que pequenos estúdios e desenvolvedores individuais introduzam inovações e ideias expressivas no mercado (Long et al., 2024). Plataformas como Steam, itch.io e consoles de jogos são facilitadores da distribuição de jogos independentes, multiplicando os canais de circulação e monetização e diversificando ainda mais a oferta de títulos disponíveis.

O processo de produção de jogos digitais é complexo e multifacetado, envolvendo uma série de etapas que vão desde a concepção inicial até o lançamento e a manutenção pós-lançamento. Esse processo é frequentemente estruturado em torno de modelos de desenvolvimento variados, como o Ciclo de Vida de Desenvolvimento de Software (SDLC), metodologias ágeis e modelos clássicos. O ciclo de vida de um jogo inclui fases de pré-produção, em que são definidos os conceitos e design iniciais; produção, na qual o desenvolvimento técnico e artístico ocorre; e pós-produção, que inclui testes, ajustes finais e suporte contínuo (Aleem et al., 2016).

A produção artística é um componente central no desenvolvimento de jogos, integrando arte visual, design de som, narrativa e mecânicas de jogo de forma coesa. De acordo com Aleem et al. (2016), essa intersecção entre dimensões artísticas e técnicas distingue o desenvolvimento de jogos do desenvolvimento de software tradicional, que tende a focar mais em funcionalidade e eficiência. No desenvolvimento de jogos, a experiência do usuário é primordial, e cada elemento do jogo deve contribuir para uma experiência imersiva e que gere entretenimento.

Nessa perspectiva, a criação de jogos digitais é um campo complexo que combina arte e tecnologia, requerendo uma abordagem multifacetada para o desenvolvimento de software (Aleem et al., 2016). Portanto, estudar os diferentes modelos de desenvolvimento de software aplicados aos jogos digitais é essencial para compreender melhor os desafios e as oportunidades que esse setor apresenta.

O presente trabalho se propõe a realizar esse estudo, por meio de uma revisão sistemática de literatura (RSL) (Grant and Booth, 2009) com o objetivo de mapear e analisar estudos sobre a aplicação de metodologias de engenharia de software para o desenvolvimento de jogos digitais. Iniciativas semelhantes foram realizadas por Osbourne-O'Hagan et al. (2014) e por Zhu and Wang (2019) na década passada. A presente revisão, compreendendo o período até maio de 2024, se soma a estas, trazendo resultados mais atuais sobre o tema. Com esta pesquisa, espera-se identificar as principais metodologias aplicadas, suas características, contexto de aplicação e desempenho. Sob o ponto de vista metodológico, este trabalho consiste em uma pesquisa exploratória (Wazlawick, 2008), de natureza básica e abordagem híbrida.

A estrutura deste artigo é composta de 5 seções: introdução; aportes teóricos (uma breve discussão a respeito de conceitos fundamentais do campo); metodologia; resultados (uma junção das métricas e constatações feitas a partir da metodologia definida) e considerações finais.

2 Aportes teóricos

O desenvolvimento de software para jogos digitais é um processo multifacetado que requer uma abordagem integrada de várias áreas do conhecimento, incluindo design, programação, arte e gestão de projetos. O ciclo de vida de desenvolvimento de software para jogos, também conhecido como Game Development Life Cycle (GDLC), é uma estrutura que descreve as fases distintas pelas quais um jogo passa desde a concepção até o lançamento e além (Ramadan and Widyani, 2013). Esse ciclo de vida pode ser dividido em etapas principais: pré-produção, produção, pós-produção e manutenção.

Na fase de pré-produção, o foco está na concepção e planejamento do jogo. Para algumas abordagens, nessa etapa o Game Design Document (GDD) é criado, incluindo detalhes de todos os aspectos do jogo, como mecânicas, história, personagens, níveis, arte conceitual e requisitos técnicos (Salazar et al., 2012). Essa fase é crucial, pois define a visão do jogo e serve como um guia para toda a equipe de desenvolvimento durante as fases subsequentes. Além do GDD, podem-se realizar nessa etapa estudos de viabilidade técnica e econômica, bem como a elaboração de protótipos iniciais para testar conceitos de jogo.

A fase de produção é quando a maioria do desenvolvimento do jogo ocorre (Aleem et al., 2016). Durante essa etapa, programadores, artistas, designers de som e outros membros da equipe trabalham em conjunto para construir o jogo conforme especificado no planejamento. Ferramentas e motores de jogo, como Unity ou Unreal Engine, são frequentemente utilizados para facilitar o desenvolvimento. A programação envolve a criação de código-fonte para implementar mecânicas de jogo, física e outros sistemas. O trabalho artístico envolve modelos 3D, texturas, animações e outros elementos visuais. O trabalho de som engloba criação de efeitos sonoros e trilhas sonoras.

A pós-produção inclui etapas como testes, depuração e ajustes finais. O Quality Assurance (QA) é uma parte crítica deste processo, garantindo que o jogo seja lançado sem bugs significativos e com uma experiência de usuário satisfatória. Testes beta são comuns nesta fase, permitindo que jogadores selecionados experimentem o jogo e deem feed-

back valioso para melhorias finais (Ramadan and Widyani, 2013). Após o lançamento, a fase de manutenção envolve a correção de bugs, atualizações de conteúdo e suporte ao jogador, garantindo que o jogo permaneça relevante e funcional ao longo do tempo.

Modelos de desenvolvimento de software são metodologias estruturadas que guiam o processo de criação de software. Entre os modelos mais conhecidos estão o cascata e o ágil. O modelo de cascata é linear e sequencial, com cada fase dependendo da conclusão da anterior. É tradicionalmente utilizado em projetos de software cujos requisitos são bem definidos desde o início (Haraty and Hu, 2018). O modelo ágil, por outro lado, é mais flexível e iterativo, permitindo ajustes constantes ao longo do desenvolvimento.

A principal diferença entre o desenvolvimento de software tradicional e o desenvolvimento de software para jogos reside na ênfase na experiência do usuário (Haraty and Hu, 2018). Enquanto o desenvolvimento de software convencional prioriza funcionalidade e outros requisitos não funcionais como eficiência, segurança e usabilidade, o desenvolvimento de jogos se concentra na criação de uma experiência imersiva e envolvente. Isso exige uma integração harmoniosa de elementos visuais, sonoros e interativos, algo que não é típico no desenvolvimento de software corporativo ou utilitário.

A produção artística também desempenha um papel fundamental no desenvolvimento de jogos. Artistas de conceito, modeladores 3D, animadores e designers de interface de usuário trabalham juntos para criar o mundo visual do jogo. A colaboração entre os departamentos de arte e programação é vital para garantir que os elementos artísticos sejam integrados eficientemente ao jogo sem comprometer o desempenho. Além disso, a organização dos processos no desenvolvimento de jogos é um desafio distinto. Isso envolve coordenar uma equipe multifuncional, gerenciando prazos e orçamentos, a fim de assegurar a qualidade final do produto.

A produção independente também integra o cenário de desenvolvimento de jogos. Desenvolvedores independentes, ou estúdios, apesar de trabalharem com recursos limitados, produzem obras importantes para o cenário. O desenvolvimento independente se destaca pela criatividade e a originalidade, muitas vezes resultando em jogos que exploram conceitos e mecânicas inovadoras (Long et al., 2024). No cenário indie, as atividades de produção de um jogo são comumente performadas por equipes pequenas e multidisciplinares, que participam em mais etapas da produção em comparação ao desenvolvimento corporativo.

Com isso, observa-se que o desenvolvimento de software para jogos é um processo complexo e interdisciplinar que combina tecnologia e arte para criar experiências interativas únicas. Entender os diferentes modelos de desenvolvimento, a importância do game design, e as particularidades da produção artística é essencial para qualquer estudo acadêmico ou prático nesta área. A constante evolução dos métodos e ferramentas de desenvolvimento reflete a natureza dinâmica e inovadora da indústria de jogos digitais.

3 Metodologia

Esta seção se reserva à descrição da metodologia empregada no processo de revisão sistemática de literatura, a fim de promover replicabilidade e transparência. O trabalho realizado apoia-se nos procedimentos descritos por Kitchenham and Charters (2007), direcionados à execução de RSLs no contexto da engenharia de software. Booth et al. (2016) compõem o norte da definição de revisão sistemática de literatura usada nesta pesquisa. Para uma organização clara, esta seção é particionada em subseções que descrevem as principais fases da revisão.

3.1 Time de revisão

Antecedendo as etapas do processo performado, faz-se necessário caracterizar a equipe envolvida no trabalho. O grupo de trabalho é composto de 3 pessoas, das quais o primeiro autor dedicou-se à elaboração e performance do processo de revisão - incluindo atividades como extração de dados, seleção de artigos, planejamento de processos e outras. O segundo autor trabalhou supervisionando o trabalho, com atividades que envolveram a verificação dos dados, administração do cronograma, revisão textual e controle de qualidade da metodologia. O terceiro autor, como especialista em Engenharia de Software, atuou na supervisão, conceituação, verificação dos dados, composição da base teórica e validação do estudo.

3.2 Questões de revisão

O planejamento do processo de revisão e de elaboração das questões para revisão seguiu o framework PICOC (Population, Intervention, Comparison, Outcome, and Context) (Kitchenham and Charters, 2007). O framework é uma ferramenta de sintetização e organização de revisões, destacando os principais elementos necessários para seu planejamento e condução (Mengist et al., 2020) (Carrera-Rivera et al., 2022). A Tabela 1 descreve os elementos do PICOC e expõe a aplicação de cada um neste estudo.

Tabela 1: PICOC

Conceito	Descrição	Aplicação
Population	Domínio de aplicação. Exemplo: Um setor industrial, uma disciplina, uma área da ciência e similares	Engenharia de Software; Desenvolvimento de Jogos digitais
Intervention	Metodologia ou ferramentas	Metodologias de engenharia de software
Comparison	Metodologias ou ferramentas para comparação	Não se aplica
Outcome	Possíveis resultados	Principais metodologias aplicadas ao desenvolvimento de jogos digitais
Context	Contexto de interesse	Desenvolvimento de jogos digitais

Baseando-se na aplicação do framework, as questões de revisão foram elaboradas para este estudo e podem ser vistas na Tabela 2:

Tabela 2: Questões de revisão

Id	Questão
QR1	Quais são os modelos de engenharia de software aplicados ao desenvolvimento de jogos digitais apresentados ou propostos pelo artigo?
QR2	O artigo propõe a adaptação ou um novo modelo de desenvolvimento de jogos?
QR3	Como esses modelos são aplicados?
QR4	Quais são as ferramentas e técnicas envolvidas?

A primeira questão de revisão (QR1) busca identificar os modelos de desenvolvimento software aplicados ao desenvolvimento de jogos. Isso é fundamental para compreender as melhores práticas e as abordagens mais eficientes utilizadas na indústria de jogos. Diferentes métodos podem ter implicações significativas na qualidade do produto final e na eficiência do processo de desenvolvimento. A questão visa catalogar modelos como cascata, ágil, e outros no desenvolvimento de jogos.

A segunda questão de revisão (QR2) investiga se o artigo propõe a adaptação de um modelo existente ou a criação de um novo modelo de desenvolvimento de jogos. Esta questão é essencial para compreender se há uma inovação ou uma melhoria significativa nos processos tradicionais de desenvolvimento de software aplicado aos jogos digitais. Avaliar a proposta de adaptação ou criação de um novo modelo permite entender as contribuições do artigo para o campo, identificando se ele traz uma nova abordagem que pode otimizar o desenvolvimento de jogos ou se sugere ajustes em metodologias já consolidadas para melhor atender às necessidades específicas de um projeto.

A terceira questão de revisão (QR3) se propõe entender como os modelos de desenvolvimento de software são aplicados no desenvolvimento de jogos. Respondê-la é crucial para identificar as adaptações e personalizações feitas para lidar com os desafios únicos da criação de jogos. Enquanto o desenvolvimento de software tradicional pode focar mais na funcionalidade e eficiência, o desenvolvimento de jogos exige uma abordagem que integre arte, narrativa e interatividade de maneira coesa (Aleem et al., 2016). Esta questão busca explorar as práticas específicas e as variações desses métodos quando implementados em projetos de desenvolvimento de jogos, destacando as melhores práticas, as lições aprendidas e os desafios enfrentados pelas equipes de desenvolvimento. A questão também levanta pontos como o tamanho das equipes de trabalho, o contexto de produção e a complexidade do produto final. Para auxiliar na clareza da questão, a Tabela 3 conta com questões de apoio à QR3.

A quarta questão de revisão (QR4) foca nas ferramentas e técnicas que suportam o desenvolvimento de jogos. Ferramentas como motores de jogo (Unity, Unreal Engine), softwares de modelagem 3D (Blender, Maya) e plataformas de design de som são cruciais para a criação de jogos. Além disso, técnicas específicas de programação, design e gestão de projetos são empregadas para garantir a coesão e a eficiência do processo de desenvolvimento. Esta questão visa identificar e catalogar essas ferramentas e técnicas.

Tabela 3: Questões de apoio à QR3

Id	Questão
QR3.1	De que tipo de jogo se trata?
QR3.2	Qual é o objetivo da produção?
QR3.3	Como se caracteriza a equipe de desenvolvimento?

3.3 Buscas

Para a busca, foram consultados os repositórios IEEE Xplore[1], ACM Digital Library[2] e SBC Open Lib (SOL)[3]. A string de busca utilizada foi: ("game development model"OR "game development models") OR (("software development model"OR "software development models") AND (games OR game)). O texto foi construído para capturar variações na terminologia, abrangendo um número maior de estudos, porém mantendo especificidade e objetividade em relação ao tema.

Cabe ressaltar que a busca foi realizada entre janeiro e maio de 2024, sem utilizar filtros especiais, tais quais filtros de tipos de artigo, intervalo temporal ou local de publicação.

3.4 Critérios de inclusão e exclusão

Ao conduzir uma revisão sistemática de literatura, é essencial estabelecer critérios claros de inclusão e exclusão (Kitchenham and Charters, 2007). Os critérios de inclusão definem os atributos específicos que os estudos devem ter para serem considerados na revisão, como o assunto relevante, o intervalo de datas de publicação, o desenho do estudo e a população estudada. Por outro lado, os critérios de exclusão delineiam as características que desqualificam os estudos de serem incluídos, como irrelevância para a questão de pesquisa, baixa qualidade metodológica ou duplicação. Para este estudo, consideramos os seguintes pontos: período, idioma, tipo da fonte, acesso e questões de revisão (Carrera-Rivera et al., 2022). Para tanto, os critérios de inclusão e exclusão utilizados estão definidos e listados na Tabela 4.

3.5 Seleção

A seleção foi realizada em duas etapas: na primeira, foi feita uma triagem por meio do título, resumo e seção de conclusão dos estudos, excluindo-se os trabalhos que não atendiam aos critérios de período, idioma, tipo de literatura, acesso e que não respondiam à questão de revisão QR1. Já na segunda etapa, realizou-se a leitura integral dos artigos provenientes da primeira etapa e a aplicação dos demais critérios de exclusão e inclusão estabelecidos. A seguir, a Tabela 5 exibe os resultados por etapa e por repositório de busca.

[1]https://ieeexplore.ieee.org/Xplore/home.jsp
[2]https://dl.acm.org/
[3]https://sol.sbc.org.br/index.php/indice

Tabela 4: Critérios de Exclusão/Inclusão

Critério	Descrição	Inclusão/Exclusão
Período	Serão incluídos estudos sem filtro de período.	**Inclusão:** Todos os períodos. **Exclusão:** Não se aplica.
Idioma	Serão incluídos apenas textos redigidos em inglês ou português. Textos em outros idiomas não serão incluídos.	**Inclusão:** Textos em inglês e português. **Exclusão:** Textos em outros idiomas.
Fontes	Serão incluídos apenas artigos científicos.	**Inclusão:** Literatura branca. **Exclusão:** Literatura cinza.
Acesso	Serão incluídos apenas textos acessíveis para membros da UFMT via parcerias, biblioteca digital, etc.	**Inclusão:** Todos, exceto itens em exclusão. **Exclusão:** Textos não acessíveis em razão de qualquer tipo de bloqueador/restrição de conteúdo
Questões de revisão	Serão incluídos apenas textos que respondam, ainda que parcialmente, às questões de revisão QR1 e QR2.	**Inclusão:** Artigos que respondem parcialmente às questões de revisão QR1 e QR2. **Exclusão:** Artigos que não respondem às questões QR1 e QR2.

A Fig. 1 descreve a seleção final de artigos (2º etapa), destacando a intersecção de artigos que foram encontrados tanto na ACM, quanto na IEEE. Por essa razão, o total apresentado na Tabela 5 não é trivialmente a soma dos itens da coluna, levando em consideração a intersecção e apresentando o número total de artigos únicos.

A Tabela 6 contém os artigos selecionados na fase final.

4 Resultados

A presente seção concentra-se na exposição de algumas métricas geradas a partir da coleta de dados e reflete as questões de revisão a partir dos textos selecionados.

4.1 QR1 - Quais são os modelos de engenharia de software aplicados ao desenvolvimento de jogos digitais apresentados ou propostos pelo artigo?

Os artigos revisados abrangem um amplo espectro de modelos de engenharia de software adaptados para o desenvolvimento de jogos digitais, ilustrando os variados contextos e objetivos que esses projetos envolvem.

Tabela 5: Critérios de Exclusão/Inclusão

Base	Resultados	1a etapa	2a etapa
ACM	253	21	7
IEEE	11	6	4
SOL	1	0	0
Total	**265**	**25**	**9**

Figura 1: Artigos da 2a etapa

No artigo "A Computer Science Study Abroad with Service Learning: Design and Reflections"(Pollock et al., 2018), a metodologia ágil foi utilizada na criação de jogos educacionais por estudantes de Ciência da Computação. O uso dessa metodologia se deu por meio do desenvolvimento de versões incrementais dos jogos, com avanços semanais. Essa abordagem possibilitou ajustes rápidos a mudanças, integrando feedback contínuo dos participantes ao longo do processo. O método focou em entregas frequentes e melhorias contínuas, resultando em versões alfa, beta e, eventualmente, finais dos jogos.

Outro modelo é destacado no artigo "Channeling End-User Creativity: Leveraging Live Streaming for Distributed Collaboration in Indie Game Development"(Li et al., 2022), que aborda o desenvolvimento independente de jogos e apresenta o modelo de desenvolvimento colaborativo. Nesse contexto, a colaboração distribuída e ferramentas digitais como o Twitch, para transmissão ao vivo, desempenharam papéis cruciais. Esse modelo ressalta a importância da flexibilidade e criatividade, permitindo que desenvolvedores e artistas colaborem remotamente e recebam feedback em tempo real da comunidade de jogadores, promovendo uma abordagem altamente iterativa e colaborativa.

Em "Developing Games That Capture and Engage Users"(Seif El-Nasr, 2019), é apresentado o modelo de desenvolvimento orientado por dados, mostrando como a coleta e análise de telemetria, métricas de comportamento dos jogadores e visualizações analíticas podem apoiar o processo de desenvolvimento de jogos digitais. A autora demonstra, por meio de estudos de caso, que esse modelo permite identificar bugs, detectar problemas de design, analisar estratégias de jogadores e avaliar a progressão de níveis, tornando possível ajustar e aprimorar o jogo de forma mais precisa e informada. Dessa forma, o artigo evidencia que o uso sistemático de dados é o elemento central para orientar decisões e otimizar a engenharia de software

Tabela 6: Seleção final de artigos

Id	Artigo
A1	"A Computer Science Study Abroad with Service Learning: Design and Reflections"(Pollock et al., 2018)
A2	"Channeling End-User Creativity: Leveraging Live Streaming for Distributed Collaboration in Indie Game Development"(Li et al., 2022)
A3	"Developing Games That Capture and Engage Users"(Seif El-Nasr, 2019)
A4	"Digital Congkak: The Art, Play and Experience Framework"(Mohamad et al., 2016)
A5	"Establishing an Educational Game Development Model: From the Experience of Teaching Search Engine Optimization"(Lui and Au, 2018)
A6	"Learning VR Game Development Towards Software Basic Profile"(Jirapanthong et al., 2017)
A7	"Managing the Development of Digital Educational Games"(Hodgson et al., 2010)
A8	"Proposing a Hybrid Methodology for Game Development"(Pandey et al., 2018)
A9	"Using Prototypes in Early Pervasive Game Development"(Ollila et al., 2008)

em jogos digitais.

O artigo "Digital Congkak: The Art, Play, and Experience Framework"(Mohamad et al., 2016) introduz o framework *Design, Play, and Experience Framework*, aplicado à criação do jogo *Digital Congkak*. Esse modelo baseia-se no conceito de *emotional design*, de Norman, especialmente no nível visceral, integrando elementos visuais, sonoros e de jogabilidade para orientar a construção da experiência do jogador. A abordagem proposta enfatiza o uso de estética, narrativa e personagens culturalmente contextualizados, de modo a preservar elementos da herança tradicional do Congkak no ambiente digital. Embora centrado na dimensão cultural e na experiência do usuário, o artigo não descreve a metodologia como uma inovação formal, mas como uma aplicação de princípios de design emocional ao contexto de jogos culturais.

O trabalho "Establishing an Educational Game Development Model: From the Experience of Teaching Search Engine Optimization"(Lui and Au, 2018) discute diversos modelos de desenvolvimento de jogos educacionais, que seguem etapas de definição de propósito, design de ideia e design de hardware e software. Contudo, a principal contribuição do artigo é a formulação de um novo modelo, denominado *Spiral Educational Game Development Model*, que integra e aprimora etapas anteriores, incorporando ciclos iterativos de projeto, prototipagem, avaliação e ajustes sucessivos. Essa abordagem baseia-se em princípios da engenharia de software, como o modelo espiral, enfatizando o refinamento contínuo e incremental dos protótipos até a obtenção de um produto educacional equilibrado entre objetivos pedagógicos e aspectos lúdicos.

O desenvolvimento de jogos de Realidade Virtual (VR) é abordado no artigo "Learning VR Game Development Towards Software Basic Profile"(Jirapanthong et al., 2017), que propõe um framework para o desenvolvimento dessa modalidade de jogo. O exemplo fornecido é o jogo *Know More Thai*, destinado a ensinar etiqueta e cultura tailandesa a estrangeiros. O framework foi projetado para criar uma experiência imersiva e educacional, adaptando as etapas tradicionais de desenvolvimento de software para atender às necessidades e desafios únicos dos jogos de VR.

Por outro lado, o artigo "Managing the Development of Digital Educational Games"(Hodgson et al., 2010) discute o *Seven-Stage Rapid Game Development Model*, um método estruturado que compreende sete etapas para o desenvolvimento de jogos educacionais. Essas etapas vão desde concepção e design até prototipagem e teste, com forte ênfase na colaboração entre as equipes de conteúdo, design e programação. Esse modelo é ideal para o desenvolvimento rápido e eficiente de jogos educacionais, especialmente em cenários em que os prazos são apertados e os recursos limitados.

O artigo "Proposing a Hybrid Methodology for Game Development"(Pandey et al., 2018) propõe um modelo híbrido de desenvolvimento de jogos que integra elementos de métodos clássicos de engenharia de software a processos criativos específicos do desenvolvimento de jogos. Esse modelo é particularmente vantajoso para projetos que demandam flexibilidade e iteração, permitindo atender simultaneamente às exigências técnicas e aos aspectos criativos do produto.

Finalmente, o artigo "Using Prototypes in Early Pervasive Game Development"(Ollila et al., 2008) apresenta a aplicação de modelos de desenvolvimento ágil e de prototipagem rápida como principais abordagens de engenharia de software voltadas ao desenvolvimento de jogos. Esses modelos enfatizam ciclos curtos de iteração que envolvem design, codificação e testes, permitindo a constante verificação da qualidade e funcionalidade do software. O texto destaca que o uso de métodos ágeis é vantajoso em jogos pervasivos por facilitar ajustes rápidos e reduzir riscos em projetos inovadores. Além do modelo ágil, o estudo também menciona o uso de prototipagem com componentes de software prontos e de prototipagem física ou em papel, que complementam o processo de engenharia ao permitir testes conceituais antes do desenvolvimento final.

4.2 QR2 - O artigo propõe a adaptação ou um novo modelo de desenvolvimento de jogos?

Vários dos artigos revisados propõem a adaptação de modelos existentes ou a criação de novos modelos de desenvolvimento de jogos, refletindo a evolução contínua das práticas de engenharia de software na indústria de jogos.

O artigo "Digital Congkak: The Art, Play and Experience Framework"(Mohamad et al., 2016) apresenta uma nova abordagem de desenvolvimento chamada *Design, Play, and Experience Framework*. O modelo combina princípios de *emotional design* e elementos culturais característicos do Congkak, integrando estética, narrativa, personagens e mecânicas de jogo. Dessa forma, a proposta busca fortalecer a experiência do jogador, valorizando aspectos culturais e sensoriais que contribuem para tornar o jogo digital coerente com sua origem tradicional. Embora não se declare como uma metodologia inovadora, a aplicação do framework evidencia uma preocupação com o alinhamento de experiência emocional e representações culturais no processo de desenvolvimento.

Outro exemplo de adaptação é encontrado no artigo "Learning VR game development towards software basic

profile"(Jirapanthong et al., 2017), que propõe um perfil básico de processo de software específico para o desenvolvimento de jogos de Realidade Virtual (Virtual Reality - VR). Esse framework é uma adaptação das etapas tradicionais de desenvolvimento para o ambiente de VR, que possui demandas e desafios únicos, como a necessidade de uma forte integração entre hardware e software e a criação de experiências imersivas. A proposta visa facilitar a produção de jogos VR educacionais, como o *Know More Thai*, garantindo que o processo de desenvolvimento seja tanto eficaz quanto eficiente, ao mesmo tempo em que atende às expectativas dos usuários em termos de imersão e interatividade.

Já o *Seven-Stage Rapid Game Development Model*, discutido em "Managing the Development of Digital Educational Games"(Hodgson et al., 2010), é uma adaptação de práticas tradicionais de engenharia de sistemas e de gestão de projetos para o contexto específico do desenvolvimento de jogos educacionais. Esse modelo foi concebido para permitir um desenvolvimento rápido e adaptável, característica essencial em ambientes educacionais onde o tempo e os recursos são frequentemente limitados. Sua estrutura enfatiza a colaboração interdisciplinar entre designers, programadores e especialistas em conteúdo, buscando alinhar os objetivos pedagógicos com a experiência de jogo.

Em "Establishing an Educational Game Development Model: From the Experience of Teaching Search Engine Optimization"(Lui and Au, 2018), há a proposição do já mencionado modelo de desenvolvimento de jogos *Spiral Educational Game Development Model*. Esse modelo foi construído a partir da observação e análise do processo de criação do jogo educacional "SEO War" e tem como base teórica modelos anteriores. A proposta difere dos modelos tradicionais ao enfatizar a natureza iterativa do processo de desenvolvimento, permitindo revisitar e ajustar etapas prévias conforme surgem novas necessidades ou problemas. O modelo propõe um ciclo contínuo de design, prototipagem, avaliação e refinamento, refletindo uma metodologia mais dinâmica e realista em comparação com abordagens lineares, além de incluir aspectos específicos do contexto educacional, como a definição de objetivos de aprendizagem e a integração entre conteúdo e mecânicas do jogo.

O artigo "Proposing a Hybrid Methodology for Game Development"(Pandey et al., 2018) integra elementos do ciclo de vida tradicional de desenvolvimento de software (SDLC) a práticas específicas do desenvolvimento de jogos (GDLC). Essa abordagem busca oferecer maior flexibilidade e capacidade de adaptação às equipes de desenvolvimento, permitindo ajustar o processo de acordo com as necessidades técnicas e criativas do projeto. O modelo híbrido mostra-se especialmente adequado para projetos complexos, nos quais é essencial responder de forma eficiente às mudanças e garantir a qualidade do produto final.

Por fim, "Using Prototypes in Early Pervasive Game Development"(Ollila et al., 2008) propõe a adaptação de métodos já existentes, aplicando-os ao contexto específico dos jogos pervasivos. Os autores defendem o uso combinado e estratégico de abordagens como a metodologia ágil, a prototipagem rápida e a prototipagem física e guiada, ajustando-as conforme o tipo de jogo, sua fase de desenvolvimento e os recursos disponíveis. Dessa forma, o estudo propõe diretrizes para a escolha do método de prototipagem adequado, considerando fatores como o propósito do protótipo, a natureza do jogo (espacial, temporal ou social) e o grau de inovação técnica. Assim, trata-se de uma evolução prática de metodologias conhecidas, voltada para contextos mais complexos e experimentais.

Essas propostas de adaptação e novos modelos ilustram como as metodologias de desenvolvimento de jogos estão sendo constantemente refinadas para melhor atender às necessidades dos desenvolvedores e às exigências específicas de diferentes tipos de jogos. A inovação nessas metodologias é crucial para o sucesso no mercado, que é altamente competitivo e dinâmico, exigindo abordagens que possam se adaptar rapidamente às novas tendências e tecnologias.

4.3 QR3 - Como esses métodos são aplicados?

A aplicação prática dos métodos de desenvolvimento de jogos digitais varia significativamente de acordo com o tipo de jogo, o objetivo da produção e a composição da equipe de desenvolvimento. No artigo "A Computer Science Study Abroad with Service Learning: Design and Reflections"(Pollock et al., 2018), o desenvolvimento ágil foi essencial para o contexto educacional, no qual o foco estava em capacitar estudantes de ciências da computação. Esse modelo permitiu que as equipes, compostas por 4 a 6 membros, seguissem um ciclo de desenvolvimento iterativo, dividido em sprints semanais. Cada sprint resultava em protótipos incrementais, que eram continuamente aprimorados com base no feedback dos instrutores e dos próprios alunos. Essa abordagem não só facilitou a adaptação às mudanças rápidas, como também proporcionou um ambiente de aprendizado dinâmico, onde os estudantes podiam experimentar diferentes soluções em um curto espaço de tempo, refletindo as demandas reais do desenvolvimento de software.

Em "Channeling End-User Creativity: Leveraging Live Streaming for Distributed Collaboration in Indie Game Development"(Li et al., 2022), o uso de plataformas de streaming ao vivo, especialmente o Twitch, desempenha um papel central na colaboração entre desenvolvedores indie. Pequenos grupos de criadores, muitas vezes distribuídos geograficamente, usam transmissões ao vivo para apresentar o andamento de seus projetos, discutir ideias e receber feedback instantâneo do público. Esse ambiente de comunicação síncrona permite que espectadores e desenvolvedores atuem como parceiros no processo criativo, contribuindo para um modelo de desenvolvimento mais democrático e participativo.

Segundo o artigo "Developing Games That Capture and Engage Users"(Seif El-Nasr, 2019), o modelo de desenvolvimento orientado por dados é aplicado integrando telemetria e análises de métricas diretamente ao processo de criação e melhoria dos jogos. Isso ocorre por meio da coleta contínua de dados de jogadores (como trajetórias, ações, erros, tempo gasto e resultados), que são então visualizados e analisados com ferramentas específicas. Essas análises permitem aos desenvolvedores identificar bugs, detectar problemas de design, avaliar o equilíbrio e a dificuldade e diagnosticar falhas na progressão dos níveis,

possibilitando ajustes fundamentados no comportamento real dos usuários. Assim, o processo de desenvolvimento se torna iterativo e guiado por evidências, com decisões tomadas com base na interpretação dos dados coletados durante o uso do jogo.

O *Spiral Educational Game Development Model*, descrito pelo texto "Establishing an Educational Game Development Model: From the Experience of Teaching Search Engine Optimization"(Lui and Au, 2018), é aplicado de forma iterativa e incremental, em ciclos que incluem as etapas de definição dos objetivos de aprendizagem, identificação das necessidades e metas do jogo, concepção conceitual, design do jogo, prototipagem, avaliação e identificação de soluções. Cada iteração gera melhorias no protótipo, com ajustes em regras, mecânicas, conteúdo e balanceamento, até que o produto final atinja os objetivos pedagógicos e de jogabilidade. No caso do jogo "SEO War", os autores seguiram essas etapas, revisando conceitos e componentes do jogo com base em observações, testes com alunos e entrevistas com os designers.

No contexto do desenvolvimento de jogos de Realidade Virtual, como exemplificado no artigo "Learning VR game development towards software basic profile"(Jirapanthong et al., 2017), a aplicação do modelo foi altamente especializada. O jogo *Know More Thai* foi desenvolvido com o objetivo de ensinar etiqueta e cultura tailandesa a estrangeiros, combinando aspectos culturais autênticos com tecnologias imersivas de VR. A proposta baseia-se em um perfil básico de processo de software voltado para o desenvolvimento de jogos, que organiza de forma sistemática as etapas tradicionais, como concepção, design, modelagem, animação e implementação, para atender às particularidades do ambiente de realidade virtual. Durante o projeto, foram realizadas observações, questionários e entrevistas para avaliar o uso desse perfil no processo de desenvolvimento, possibilitando o aperfeiçoamento da experiência educacional e imersiva oferecida pelo jogo.

Em "Using Prototypes in Early Pervasive Game Development"(Ollila et al., 2008), os modelos são aplicados por meio de prototipagem iterativa nas fases iniciais de desenvolvimento, visando testar e validar ideias de design antes da implementação completa. O método ágil é utilizado para desenvolver protótipos funcionais em ciclos curtos, permitindo ajustes contínuos conforme o feedback dos testes. A prototipagem com softwares prontos, como fóruns e motores de jogos, é empregada para simular mecânicas e interações em ambientes virtuais. Já a prototipagem física e guiada utiliza materiais simples, como papel, miniaturas e planilhas, para simular jogabilidade e coletar observações sobre usabilidade e comportamento dos jogadores.

4.4 QR4 - Quais são as ferramentas e técnicas envolvidas?

As ferramentas e técnicas utilizadas no desenvolvimento de jogos digitais são tão diversas quanto os próprios métodos aplicados. As ferramentas e técnicas utilizadas no desenvolvimento de jogos digitais, segundo o artigo "A Computer Science Study Abroad with Service Learning: Design and Reflections"(Pollock et al., 2018), foram selecionadas para apoiar um processo ágil e iterativo. Os estudantes utilizaram HTML5 com o framework Phaser e o Unreal Engine como principais plataformas de desenvolvimento, criando protótipos jogáveis que evoluíam ao longo de ciclos semanais de desenvolvimento.

No artigo "Channeling End-User Creativity: Leveraging Live Streaming for Distributed Collaboration in Indie Game Development"(Li et al., 2022), as plataformas de live streaming emergem como a principal ferramenta utilizada pelos desenvolvedores indie para apoiar a colaboração distribuída. O streaming em tempo real funciona simultaneamente como ambiente de trabalho, espaço de co-design e meio de comunicação síncrona, permitindo que criadores compartilhem seu progresso, discutam decisões de design e obtenham feedback instantâneo do público. Essas transmissões também servem como um ponto de encontro social, no qual espectadores e potenciais jogadores podem participar do processo criativo, sugerir soluções, avaliar funcionalidades e contribuir para a construção coletiva da identidade e da visibilidade dos projetos.

No modelo apresentado pelo artigo "Developing Games That Capture and Engage Users"(Seif El-Nasr, 2019), a produção de jogos envolve principalmente ferramentas e técnicas de análise de dados, utilizadas para orientar decisões de design e identificar problemas no desenvolvimento. Entre essas técnicas, estão a coleta de telemetria do comportamento dos jogadores, o cálculo de métricas agregadas para sintetizar grandes volumes de dados e o uso de visualizações especializadas, como heatmaps para representar informações espaço-temporais e painéis comparativos que distinguem trajetórias de jogadores que obtiveram sucesso ou fracasso.

O desenvolvimento em "Establishing an Educational Game Development Model: From the Experience of Teaching Search Engine Optimization"(Lui and Au, 2018) abordou diversas ferramentas e técnicas de apoio ao processo iterativo. Entre elas, destacam-se o uso de modelos causais e representações gráficas para visualizar relações entre conceitos de aprendizagem e atributos do jogo, planilhas consolidadas para organizar informações sobre cartas e mecânicas, além de protótipos físicos. Também foram descritas técnicas de coleta de dados, como observação de reuniões de design, questionários aplicados e entrevistas. Além disso, destacam-se outros elementos de engenharia de software, como controle de versões de protótipos e testes de usabilidade.

No desenvolvimento de jogos de Realidade Virtual, conforme descrito em "Learning VR Game Development Towards Software Basic Profile"(Jirapanthong et al., 2017), ferramentas como Unity 3D e 3Ds Max desempenharam papéis centrais na criação de ambientes virtuais imersivos. O Unity 3D foi utilizado para o desenvolvimento da estrutura e integração do jogo, permitindo a implementação dos modelos tridimensionais, sons e demais elementos interativos. Já o 3Ds Max foi empregado na modelagem dos personagens e cenários, com base em proporções reais e estudo de campo, o que contribuiu para a fidelidade visual e cultural do projeto.

No artigo "Managing the Development of Digital Educational Games"(Hodgson et al., 2010), o uso do Seven-Stage Rapid Game Development Model envolveu um conjunto estruturado de práticas destinadas a acelerar o desenvol-

vimento de jogos educacionais baseados na web. O processo foi apoiado por documentação detalhada, incluindo storyboards e descrições de design que orientavam a implementação técnica. As equipes, organizadas em áreas de conteúdo, design e programação, mantinham comunicação contínua e realizavam revisões regulares para garantir a coerência entre os objetivos pedagógicos e os resultados do jogo.

Por fim, as ferramentas e técnicas descritas no artigo "Using Prototypes in Early Pervasive Game Development"(Ollila et al., 2008) incluem o uso de plataformas de prototipagem rápida, como o MUPE (Multi-User Application Platform). Também são mencionadas ferramentas de software reutilizáveis, como fóruns web, e o uso de métodos físicos de prototipagem, incluindo protótipos em papel, simulações com objetos e o método *Wizard-of-Oz*, no qual um humano simula o comportamento do sistema. Além disso, técnicas de testes com usuários reais e grupos focais foram empregadas para coletar dados sobre atitudes, opiniões e comportamento dos jogadores.

5 Considerações finais

A análise dos artigos selecionados revela que, embora existam paralelos significativos com o desenvolvimento de software tradicional, o desenvolvimento de jogos digitais apresenta desafios e demandas únicas que requerem metodologias adaptadas e específicas. Essa distinção se torna evidente na maneira como os modelos de engenharia de software são aplicados e modificados para atender às necessidades específicas do setor de jogos, que combina de forma intrínseca aspectos técnicos e criativos.

Tal especificidade se nota, por exemplo, em frameworks como o *Design, Play, and Experience Framework*, que exemplificam a tentativa de unir design conceitual e desenvolvimento emocional para criar experiências de jogo imersivas e culturalmente significativas.

Os modelos de engenharia de software utilizados no desenvolvimento de jogos digitais variam amplamente, desde abordagens mais tradicionais até metodologias ágeis. A escolha do modelo adequado é influenciada por diversos fatores, incluindo o tamanho da equipe, o tipo de jogo em desenvolvimento, o objetivo final do projeto e as restrições de tempo e recursos disponíveis. Em projetos menores, como os de jogos independentes, observa-se uma preferência por métodos ágeis que permitem iteração rápida e adaptação contínua, enquanto, em projetos de maior escala, modelos mais estruturados podem ser preferidos para garantir consistência e qualidade ao longo de um ciclo de desenvolvimento mais extenso.

Uma tendência manifesta nos artigos analisados nesta revisão é a significativa adoção de metodologias ágeis no desenvolvimento de jogos. Essa preferência reflete a necessidade de flexibilidade e adaptação contínua em um mercado que demanda inovação constante e rápida resposta às mudanças nas expectativas dos jogadores. Modelos colaborativos, especialmente em contextos de desenvolvimento indie, demonstram como as ferramentas digitais modernas têm transformado a maneira como as equipes de desenvolvimento trabalham. Plataformas de diversas naturezas, como o Twitch, por exemplo, permitiram novas formas de coordenação e colaboração, com equipes distribuídas geograficamente operando de maneira eficaz e integrada.

Além disso, há inovações recentes que ainda não estão presentes nos trabalhos analisados nesta revisão, mas que devem ser levadas em consideração em investigações futuras sobre o tema. Isso ocorre principalmente porque, no contexto do desenvolvimento de software, incluindo jogos, os artigos científicos não são o tipo predominante de documentação. Em muitas áreas desse campo, a documentação técnica é amplamente composta por relatórios industriais, white papers, tutoriais e até mesmo postagens em blogs especializados, que frequentemente são mais céleres em manifestar tendências.

Como exemplo dessas inovações, as inteligências artificiais generativas têm desempenhado um papel cada vez mais significativo no desenvolvimento de software e, em especial, na criação de jogos digitais. Essas tecnologias permitem automatizar etapas complexas, como a geração de código, a criação de assets visuais e sonoros, o design de níveis e até o protótipo de mecânicas de jogos.

Dessa forma, a inclusão de inovações como as inteligências artificiais generativas nas análises futuras é essencial para compreender não apenas o estado atual do desenvolvimento de software e jogos, mas também a direção para a qual essas práticas estão evoluindo.

Em suma, o campo do desenvolvimento de jogos digitais continua a evoluir, impulsionado pela necessidade de metodologias que não só atendam aos requisitos técnicos, mas também integrem os elementos criativos e artísticos que definem a experiência de jogo. A adaptabilidade dos modelos de engenharia de software a esses requisitos específicos é crucial para o sucesso no desenvolvimento de jogos, pois se trata de ambientes complexos e extremamente contextuais. Nesse sentido, estudos como esse realizado podem apoiar equipes de desenvolvimento na seleção das metodologias e tecnologias mais adequadas para a engenharia de jogos.

Agradecimentos

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro fornecido por meio de bolsa de iniciação científica e bolsa produtividade em pesquisa, que foram essenciais para o desenvolvimento desta pesquisa. Ainda, agradecem ao projeto Dados Além da Vida (DAVI) do Instituto de Computação da UFMT pelas constantes reflexões interdisciplinares.

Referências

Aleem, S., Capretz, L. F. and Ahmed, F. (2016). Game development software engineering process life cycle: a systematic review, *Journal of Software Engineering Research and Development* 4: 1–30. https://doi.org/10.1186/s40411-016-0032-7.

Booth, A., Sutton, A. and Papaioannou, D. (2016). *Systematic Approaches to a Successful Literature Review*, 2nd edn, Sage, Los Angeles, CA, USA.

Carrera-Rivera, A., Ochoa, W., Larrinaga, F. and Lasa, G. (2022). How-to conduct a systematic literature review: A quick guide for computer science research, *MethodsX* **9**: 101895. https://doi.org/10.1016/j.mex.2022.101895.

Grant, M. and Booth, A. (2009). A typology of reviews: an analysis of 14 review types and associated methodologies, *Health Information & Libraries Journal* **26**: 91–108. https://doi.org/10.1111/j.1471-1842.2009.00848.x.

Haraty, R. A. and Hu, G. (2018). Software process models: a review and analysis, *International Journal of Engineering & Technology* **7**(2.28): 390–397. https://doi.org/10.14419/ijet.v7i2.29.13206.

Hodgson, P., Man, D. and Leung, J. (2010). Managing the development of digital educational games, *2010 Third IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning*, pp. 191–193. https://doi.org/10.1109/DIGITEL.2010.18.

Jirapanthong, W., Yampray, K. and Tancharoen, D. (2017). Learning vr game development towards software basic profile, *2017 10th International Conference on Ubi-media Computing and Workshops (Ubi-Media)*, pp. 1–5. https://doi.org/10.1109/UMEDIA.2017.8074128.

Kitchenham, B. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering, *Technical report*, Technical Report EBSE-2007-01.

Li, L., Freeman, G. and McNeese, N. J. (2022). Channeling end-user creativity: Leveraging live streaming for distributed collaboration in indie game development, *Proc. ACM Hum.-Comput. Interact.* **6**(CSCW2). https://doi.org/10.1145/3555173.

Long, S., Denisova, A. and Mirza-Babaei, P. (2024). From pixels to play: Opportunities and challenges of a diverse and democratized games industry, *Interactions* **31**(2): 54–58. https://doi.org/10.1145/3647646.

Lui, R. W. and Au, C. H. (2018). Establishing an educational game development model: From the experience of teaching search engine optimization, *Int. J. Game-Based Learn.* **8**(1): 52–73. https://doi.org/10.4018/IJGBL.2018010104.

Marcos, A. and Zagalo, N. (2011). Instantiating the creation process in digital art for serious games design, *Entertainment Computing* **2**(2): 143–148. https://doi.org/10.1016/j.entcom.2010.12.003.

Mengist, W., Soromessa, T. and Legese, G. (2020). Method for conducting systematic literature review and meta-analysis for environmental science research, *MethodsX* **7**: 100777. https://doi.org/10.1016/j.mex.2019.100777.

Mohamad, A. F. H., Supian, J. J. M. and Ribuan, M. S. M. (2016). Digital congkak: The art, play and experience framework, *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*, IMCOM '16, Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/2857546.2857636.

Ollila, E. M. I., Suomela, R. and Holopainen, J. (2008). Using prototypes in early pervasive game development, *Comput. Entertain.* **6**(2). 10.1145/1371216.1371220.

Osbourne-O'Hagan, A., Coleman, G. and O'Connor, R. (2014). Software development processes for games: a systematic literature review, *21st European Conference on Systems, Software and Services Process Improvement EuroSPI*. https://doi.org/10.1007/978-3-662-43896-1_16.

Pandey, J., Singh, A. V. and Alabri, A. A. (2018). Proposing a hybrid methodology for game development, *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 142–146. https://doi.org/10.1109/ICRITO.2018.8748360.

Pollock, L., Atlas, J., Bell, T. and Henderson, T. (2018). A computer science study abroad with service learning: Design and reflections, *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, SIGCSE '18, Association for Computing Machinery, New York, NY, USA, p. 485–490. https://doi.org/10.1145/3159450.3159589.

Ramadan, R. and Widyani, Y. (2013). Game development life cycle guidelines, *2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, IEEE, pp. 95–100.

Read, S. (2022). Gaming is booming and is expected to keep growing. this chart tells you all you need to know. https://www.weforum.org/agenda/2022/07/gaming-pandemic-lockdowns-pwc-growth/.

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

Seif El-Nasr, M. (2019). Developing games that capture and engage users, *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, pp. 9–10. https://doi.org/10.1109/ICSE-Companion.2019.00025.

Wazlawick, R. (2008). *Metodologia de pesquisa para Ciência da Computação*, Elsevier, Rio de Janeiro.

Zhu, M. and Wang, A. I. (2019). Model-driven game development: A literature review, *ACM Computing Surveys (CSUR)* **52**(6): 1–32. https://doi.org/10.1145/3365000.