Play Your Protocol: Structuring the Development of Clinical-Protocol-Based Serious Games

Caroline da Conceição Lima
Programa de Engenharia de Sistemas e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
caroline@cos.ufrj.br

Daniel Martins
Programa de Engenharia de Sistemas e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
dsm@cos.ufrj.br

Geraldo Xexéo
Programa de Engenharia de Sistemas e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
xexeo@ufrj.br

Ryan Dias
Programa de Engenharia de Sistemas e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
ryandab@ic.ufrj.br

Samuel Araújo
Programa de Engenharia de Sistemas e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
samuelvitor86@hotmail.com

Thomas Cardoso
Programa de Engenharia de Sistemas e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
thomascard.20221@poli.ufrj.br

Antônio Pena
Instituto do Coração Edson Saad
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
antoniopenamaciel@gmail.com

Gláucia Maria Moraes de Oliveira
Instituto do Coração Edson Saad
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
glauciamoraesoliveira@gmail.com

Julia Marques
Instituto do Coração Edson Saad
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
juliamarques1406@gmail.com

Mariara Marques
Instituto do Coração Edson Saad
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
mariaramarques@ufrj.br

Paolo Blanco Villela
Instituto do Coração Edson Saad
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
pbvillela@gmail.com

Thais Salim
Instituto do Coração Edson Saad
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
thais.salim@yahoo.com.br

Camila Lopes
Escola de Belas Artes
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
camila.mqslopes@gmail.com

José Gomes
Escola de Belas Artes
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
jalexandre.gb@gmail.com

Ana Maria Tavares Cavalcanti
Escola de Belas Artes
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
ana.canti@eba.ufrj.br

Tadeu Moreira de Classe
Programa de Pós-Graduação em Informática - PPGI
Universidade Federal do Estado do Rio de Janeiro – UNIRIO
Rio de Janeiro, Brasil

Abstract

Serious medical games must translate safety-critical domain knowledge into requirements, art assets, interaction mechanics, and executable software, yet their development often relies on repeated negotiation among physicians, game designers, artists, and developers. This paper presents Play Your Protocol (PYPr), a digital game design method for serious visual-novel games based on clinical protocols. The Action Design Research (ADR) methodology proposed by Sein et al. (2011) was adopted as the main research strategy, given its support for continuous intervention and evaluation of the artifact throughout its construction. Across an exploratory cycle and three refinement cycles, the process evolved from a direct transfer of a BPMN-oriented game design method to a workflow centered on domain-native artifacts, storyboards, an Art Guideline, and staged expert validation. The final method was evaluated with nine participants, and open responses were analyzed using content analysis, producing 72 recording units and 82 coded occurrences. Clinical cases emerged as the most frequently used bridge between medical knowledge and development decisions, while intermediate validation improved alignment but remained vulnerable to delayed expert feedback. The contribution is process knowledge for game software engineering: domain-native boundary artifacts, explicit visual-fidelity requirements, and intermediate validation loops can structure interdisciplinary development, while expert-response time must itself be treated as a process dependency.

Keywords

Game Development Process, Serious Games, Software Engineering, Clinical Protocols, Medicine Games

1 Introduction

Developing a serious game from a clinical protocol is not simply a matter of placing medical content inside a game. The protocol must be transformed into player-facing situations, decision points, failure conditions, feedback, dialogue, visual evidence, mechanics, and executable state transitions [14], while preserving clinically relevant distinctions and avoiding representations that could lead to incorrect learning [16, 22].

Beyond the educational challenges, developing serious games in safety-critical domains is fundamentally a software engineering problem: clinical knowledge must be progressively translated into requirements, interaction logic, visual assets, executable behavior, and validation artifacts while preserving domain correctness [17], a translation that introduces challenges of requirements communication, artifact coordination, traceability, and quality assurance across stakeholders with different backgrounds [11]. Team composition amplifies this difficulty, medical experts do not share software engineers' modeling vocabulary, while artists and developers need concrete, implementable representations of symptoms, procedures, and behavior [13, 20]. Collaborative frameworks acknowledge the importance of multidisciplinary work [10], and medical-game studies report the value of team members who can bridge medicine and software development [1]; however, the existence of collaboration does not by itself define how domain knowledge should move through the development lifecycle [7], motivating the software process this paper proposes.

This paper investigates how to structure the development of a serious game based on a clinical protocol so that multidisciplinary teams can translate domain knowledge into software and game assets while preserving fidelity and controlling rework. We address the problem through Play Your Protocol (PYPr), a method created and refined during the development of a visual-novel serious game for Acute Coronary Syndrome (ACS) called [OMITTED FOR ANONYMOUS REVIEW]. The game itself, its clinical cases, and its educational adequacy are not the contributions of this paper; the contribution is the development process and the empirical knowledge obtained from its evolution and evaluation. The study was guided by two research questions:

RQ1: How can the development process of clinical-protocol-based serious games be structured for multidisciplinary teams?

RQ2: Which artifacts and process mechanisms most support development, and which dependencies remain problematic?

The study used Action Design Research (ADR) [19]. The problem formulation combined two rapid reviews, a survey of 15 game development participants, and a case study of a previous medical game project. The method was then built, used, evaluated, and refined across an exploratory cycle and three development cycles, with a final evaluation involving nine participants and qualitative content analysis. The resulting evidence indicates that clinical cases became the central translation artifact, that visual accuracy needs explicit requirements rather than ad hoc consultation, and that intermediate expert validation loops support quality control but create a process dependency on expert response time.

2 Related Work and Research Gap

Clinical protocols and business processes share some structural characteristics: they define roles, activities, decisions, alternatives, and expected outcomes. Clinical work, however, is more strongly shaped by patient state, uncertainty, time pressure, exceptions, and consequences of error [8, 18]. This resemblance motivated the initial attempt to transfer Play Your Process (PYP), a method that derives game design elements from business-process models [5], to medical game development. PYP guides designers through five stages, from a context study of the business process, through BPMN-to-game Elements Mapping producing an initial GDD, Game Project, Development/Prototyping, to a three-tier Evaluation involving the design team, process executors, and the target audience [4, 5]. Its central mechanism, a systematic mapping from process-model elements to game-design elements, was later automated by a companion tool, ProModGD [3].

The transfer was not straightforward because a representation useful for process engineers is not necessarily the best communication artifact for physicians, artists, and game developers, extending the challenge beyond process modeling to the systematic translation of domain knowledge into artifacts that can be understood, refined, and validated throughout the software lifecycle. Existing work approaches this challenge from different perspectives but focuses on isolated phases of the process: task-modeling approaches such as ConcurTaskTrees represent emergency procedures through executable task structures [21]; requirements-oriented methods formalize educational needs before implementation [6]; and design frameworks derive principles for simulation games or organize

multidisciplinary collaboration [1, 10, 12]. None of these explicitly define how domain knowledge encoded in clinical protocols evolves across software development artifacts throughout the lifecycle, limiting guidance on artifact traceability, staged validation, and requirements communication between domain experts and development teams.

Therefore, the missing contribution is not another game design methodology, but a software process that systematically translates domain knowledge into software artifacts throughout the development lifecycle. PYPr addresses this gap by proposing an artifact-based software development process in which clinical knowledge is progressively transformed into explicit development artifacts, validated through iterative quality gates, and communicated across multidisciplinary teams, contributing reusable software engineering knowledge on artifact coordination, knowledge translation, traceability, and quality assurance for serious game development.

3 Research Method

3.1 Action Design Research

Action Design Research (ADR) was selected because the research artifact was a development method refined through use in a real multidisciplinary project, and, from a software engineering perspective, was particularly appropriate given that the proposed contribution is itself a software process [9]. ADR combines artifact construction with intervention and evaluation in the organizational setting, allowing the artifact and the researchers' understanding of the problem to evolve together [19]: each development cycle generated process modifications that were immediately applied, evaluated, and refined, allowing the resulting process knowledge to emerge incrementally from practice.

The study followed the four ADR stages. Problem Formulation used literature evidence, a practitioner survey, and an exploratory case study. Building-Intervention-Evaluation occurred during successive development cycles. Reflection and Learning followed each cycle: in weekly meetings involving the first author, the medical coordinators, and team leads from art and development, the team reviewed the coordination problem observed during that cycle and agreed on the corresponding process change before the next cycle began. Formalizing learning produced the final PYPr process and the transferable lessons discussed in this paper.

The final version of PYPr, formalized in Cycle 3, defines four roles with explicit handoffs. The Medicine Team produces three domain-native artifacts (Care Guideline, Clinical Pathways, Clinical Cases) and, jointly with the Game Design Team, the Art Guideline. The Game Design Team consumes these to produce and update the GDD, Storyboard, and minigame specifications. The Art Team consumes the Storyboard and Art Guideline to produce visual assets, which pass through a medical validation gate (output: an Evolution Report) before implementation. The Development Team consumes validated design and art artifacts to produce a playable version, which passes through two sequential validation gates: Game Design Team validation, then Medicine Team validation, each capable of triggering revision loops back to the Storyboard, minigames, art, or GDD.

3.2 Preliminary Work

The problem-formulation studies provided complementary evidence. The first rapid review examined process models in game design and searched ACM Digital Library, El Compendex, Google Scholar, IEEE Digital Library, ISI Web of Science, Science Direct, and Wiley & Sons, 201 records screened, 15 accepted; PICOC-structured protocol [15]. The second focused on clinical protocols as game-design inputs and additionally searched Cochrane Library, ERIC, and PubMed, 763 records screened, 6 accepted.

The survey used convenience sampling (n=15) via WhatsApp groups associated with SBGames and UFRJ's LUDES laboratory, finding a tension between the perceived benefits of process models and fears of rigidity: 80% supported new methods based on process models, while 38.1% of the coded limitations concerned the possible restriction of creative expression. An explanatory and descriptive case study of a previous medical game project triangulated interviews, documents, repository history, gameplay logs, and team communication, finding an essentially ad hoc process, repeated clarification requests, structural redesign, tool migration, and difficulty translating rigorous medical documents into game logic.

Full protocols, search strings, and the survey instrument are provided as supplementary material at https://github.com/LimaCarolineC/SE4Games2026.git.

3.3 Project Context and Units of Analysis

PYPr was built during a one-year project that developed a visual-novel serious game for ACS training called SOS SCA - Síndrome Coronariana Aguda, involving medical, game design, art, and development roles. We applied the method to three clinical cases of increasing complexity, each becoming a development unit through which we could test and refine the process (Figure 1 presents the game's initial scene, showing the characters representing each clinical case). The unit of analysis for RQ1 is the evolving development method across cycles, and for RQ2 is the use and perception of its artifacts and coordination mechanisms by project participants. The final evaluation included all nine participants in the development team: two from the art team, four from development, two medical students, and one participant from the earlier medical game project providing a contrasting perspective. An institutional research ethics committee approved the study (approval no. 7178783).

3.4 Data Collection and Analysis

Evidence was collected throughout development through meeting records and field notes, design artifacts, evolving process models, development documents, direct observation, and validation feedback. The final evaluation used an online questionnaire with Likert items and open questions, followed by qualitative interpretation; because the sample was small, Likert responses were used primarily to guide interpretation rather than to support inferential claims.

Open responses and interview material were analyzed using the Content Analysis methodology proposed by Laurence Bardin [2]: the material was organized, segmented into recording units by semantic meaning, coded, and grouped into subcategories and thematic categories, yielding a final corpus of 72 recording units (some expressing more than one relevant idea and receiving more

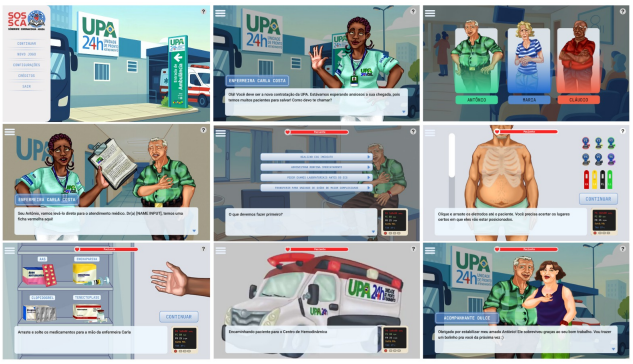

Figure 1: Scenes of the game SOS SCA. This is part of the promotional artwork for SOS-SCA, created by José Alexandre and Camila Lopes

than one code, producing 82 coded occurrences). Coding followed an inductive codebook developed by the first author during material exploration, recording each code's definition, inclusion criteria, and a representative excerpt, and is provided as supplementary material. Coding and categorization were performed by a single researcher without independent double-coding or a formally computed inter-coder agreement measure, acknowledged here as a limitation. The analysis was designed to identify what participants perceived as useful, problematic, or unnecessary in the method and to relate those perceptions to observed changes across development cycles.

4 Evolution of the Development Process

PYPr did not emerge as a finished method. It evolved through failures and bottlenecks that exposed the limitations of a generic process-to-game approach in clinical game development. Table 1 summarizes the evolution.

Each development cycle is reported as a refinement of the software process, exposing a coordination problem and motivating a process modification, as shown in Table 1.

4.1 Cycle 0 of the Play Your Protocol Method Development

Cycle 0 began with a direct application of PYP: the medical and computing students modeled the protocol in BPMN, and the team created a GDD; CMMN was also considered because patient evolution resembles case-based work. The experiment exposed two mismatches: formal process notation was familiar to software engineers but not to the medical team, introducing friction at the start of collaboration, and neither BPMN nor CMMN adequately represented all information needed by the game, including patient-specific presentation, time pressure, pedagogical feedback, graduated failure, visual symptoms, and art constraints. This cycle produced the first design principle: the method should begin from artifacts native to the domain experts and introduce formal modeling only when it demonstrably supports a production role, a substantive shift from treating notation as the center of the method to treating translation across roles as its center.

4.2 Cycle 1 of the Play Your Protocol Method Development

Cycle 1 reorganized the input around three complementary medical artifacts: the care guideline provided evidence-based recommendations, clinical pathways outlined procedural flow, and clinical cases instantiated the protocol in concrete patient situations, while the storyboard became the primary reference for the visual novel's flow and the GDD remained a project-level repository of objectives and technical decisions. The first playable version was produced during this cycle, but the main process failure appeared at validation: clinically experienced reviewers identified visual details, such as hand position, skin tone, facial expression, accessories, equipment placement, and examination imagery, that could suggest a different condition or misrepresent severity. The lesson was that visual fidelity had to be treated as a requirement and quality-assurance concern, not as a final aesthetic check.

4.3 Cycle 2 of the Play Your Protocol Method Development

Cycle 2 introduced an intermediate medical validation step for art assets and explicit loops for revision before implementation. The game also became procedurally richer through minigames, creating new dependencies among clinical rules, interface design, art, and code: a drag-and-drop ECG-electrode mechanism required medically correct positioning and legible interface design, while another scene required an ECG image with a particular abnormality in specific leads, and art production stalled when annotated references arrived late. Intermediate validation reduced the risk of incorrect assets propagating into later versions, but exposed a second-order problem: validation can only be timely if the required evidence is available, so the method needed to front-load visual reference material alongside narrative and flow requirements.

4.4 Cycle 3 of the Play Your Protocol Method Development

The final method added an Art Guideline produced early in the cycle, consolidating visual and representational constraints identified by the medical team: reference ranges for visible patient conditions, facial expression scales for pain, annotated examples of equipment and examination results, environmental references, and explicit restrictions on details that can compromise clinical accuracy. Figure 2 summarizes the final process logic: domain artifacts feed design, design artifacts feed art and development, art is validated before implementation, and a playable version is validated first by the game-design team and then by the medical team, with each validation capable of triggering revisions to storyboards, minigames, art, or the GDD.

The central structural difference from the initial PYP transfer is that the final method triangulates multiple domain artifacts and distributes translation work across specialized intermediate artifacts, rather than relying on a one-to-one mapping between a formal process model and game elements; it also separates validation of representations from validation of the executable game, recognizing that defects can originate in content interpretation, art production, interaction design, or implementation. Explicit documentation of

Table 1: Evolution of PYPr across development cycles.

Cycle	Main Process Configuration	Problem Observed	Change Introduced
0	Direct PYP experiment; BPMN/CMMN as protocol representations; GDD as central document.	Formal notation caused friction; flow omitted case, art, time pressure, feedback, and failure-grading data.	Replace notation with domain-native inputs (care guideline, pathways, cases); storyboard as implementation artifact.
1	Clinical artifacts + GDD + storyboard; first playable case; final validation.	Late review found meaningful visual errors after art and implementation were done.	Add intermediate medical validation of art assets and refinement loops.
2	Intermediate art validation; minigame design/update tasks; stronger storyboard–GDD sync.	Art production stalled awaiting references for equipment, exams, symptoms, and visual details.	Add early Art Guideline with references, constraints, and annotated examples.
3	Domain-native inputs, Art Guideline, storyboard/minigame iteration, art, design, and medical validation before release.	Method stabilized; evaluation showed remaining schedule/response-time dependencies.	Future: feedback windows, escalation paths, async review, selective notation reuse.

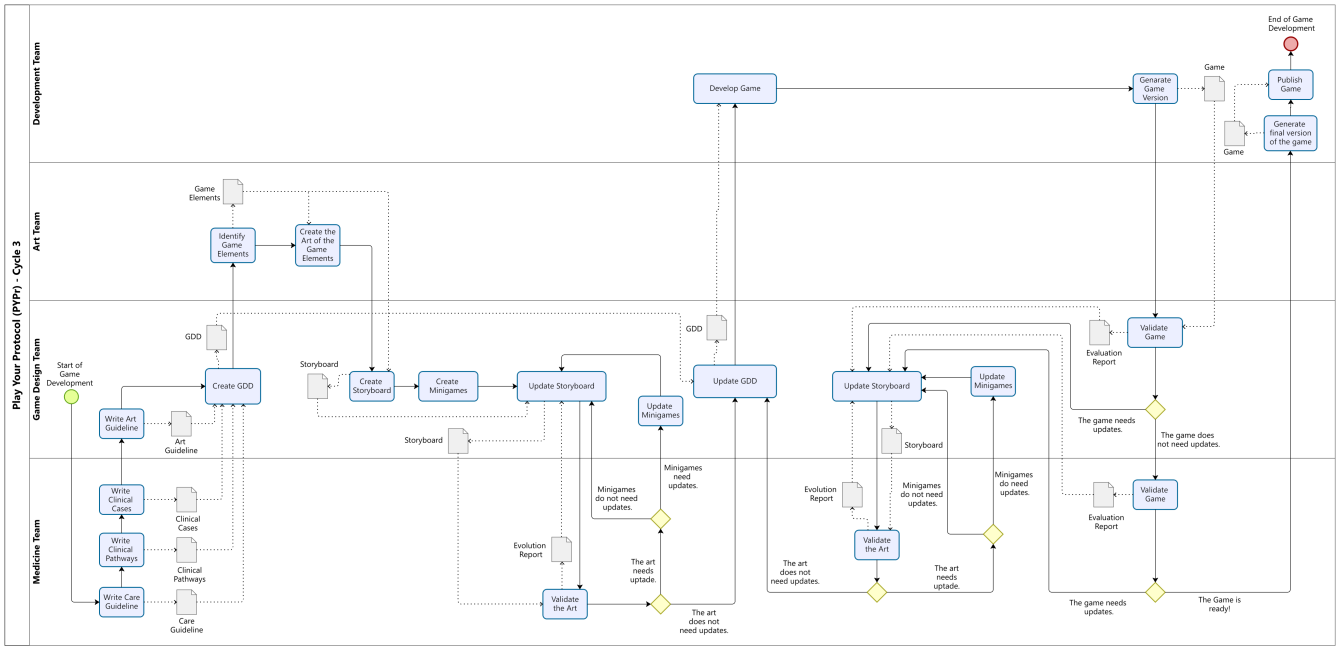

Figure 2: Representation of the final PYPr workflow. Domain-native medical artifacts feed design. The process includes an intermediate art-validation loop and two sequential validation steps for the playable version. Created by the author.

visual requirements through the Art Guideline reduced ambiguity between medical experts, artists, and developers, demonstrating the value of domain-specific artifacts in multidisciplinary software projects.

5 Evaluation Results

The final evaluation generated 72 recording units and 82 coded occurrences. Table 2 presents the five categories. The frequencies are descriptive indicators of emphasis in the qualitative corpus; they are not statistical effect measures.

Participant feedback consistently highlighted three process characteristics: improved communication through explicit artifacts, earlier detection of inconsistencies through staged validation, and better coordination among multidisciplinary roles. Rather than evaluating only the educational value of the resulting game, these findings provide empirical evidence regarding the effectiveness of the proposed software process.

5.1 Clinical Cases Became the Main Boundary Artifact

The strongest result was the centrality of clinical cases. Within the artifact category, the code "primary source of understanding"

Table 2: Content-analysis categories in the final evaluation. AF = Absolute Frequency (count of coded occurrences).

Category	AF	Main Evidence
Method Artifacts as Cognitive Facilitators	23	Clinical cases dominant; guidelines useful; BPMN ambivalent
Interdisciplinary Communication	20	Alignment improved; delayed expert feedback caused rework
Structural Support for Development	15	Clearer stages, task guidance, common initial baseline
Efficiency and Product Impact	13	Perceived delay reduction, clarity, fidelity and consistency
Limitations and Areas for Improvement	11	Schedule management, terminology, tool underuse, minigame complexity

associated with clinical cases appeared 10 times, compared with three positive and four negative/underuse occurrences for BPMN; participants described cases as directly useful for understanding the protocol, creating art, and guiding implementation, as one participant put it, “I relied more on clinical cases than on BPMN.” Guidelines and pathways added standardization and broader structural context, but cases converted abstract protocol logic into situations that all teams could reason about. Formal notation was not completely rejected, its value was role-dependent and contingent on facilitation, so the transferable lesson is not "never use BPMN" but that a process notation should not be assumed to function as a universal boundary object: in this project, clinical cases were more consistently used across medicine, art, design, and development.

5.2 Communication Improved but Expert Availability Remained a Bottleneck

Interdisciplinary Communication accounted for 20 coded occurrences: nine concerned communication and alignment, three concerned interdisciplinary validation, and eight concerned delays and rework. Participants valued shared artifacts and cross-team visibility, especially the ability to validate art before it became embedded in a playable build, but delayed medical feedback was the most persistent operational weakness, with participants citing “the response time of the doctors” as the main obstacle. PYPr structures what must be validated and when, but the evaluated version does not guarantee how quickly external experts will respond; in a dependency-sensitive pipeline, an undefined review latency behaves like an unbounded service time, so the method needs explicit response windows, escalation paths, or asynchronous review mechanisms to reduce schedule risk systematically.

5.3 Participants Perceived Structural and Product Benefits

Structural Support for Development accounted for 15 occurrences, with participants emphasizing that the initial documents and workflow clarified stages and guided actions. Efficiency and Product Impact accounted for 13 occurrences, including a perceived reduction in delays, improved fidelity and consistency, and reduced ambiguity; these are participant perceptions rather than controlled productivity measurements, but they align with the observed evolution of the process, which progressively moved validation earlier, made dependencies explicit, and created artifacts targeted at specific handoffs. A contrastive account from the participant who had worked on the earlier, unstructured project associated that experience with greater disorganization, more errors, and lower quality, indirect but suggestive evidence that the structured process was useful here.

6 Discussion

The first research question (RQ1) asked how the development process can be structured. The evidence suggests that the essential structure is a chain of translation and verification rather than merely a sequence of production activities, whose main advance is not more documentation but the allocation of distinct uncertainty classes to distinct artifacts and verification points: flow uncertainty to pathways and storyboards, decision uncertainty to clinical cases, visual-fidelity uncertainty to the Art Guideline and art validation, interaction uncertainty to minigame iteration, and integration uncertainty to playable-version validation. This distribution makes sources of defect and rework more traceable.

The second research question (RQ2) asked which artifacts and mechanisms most support or hinder development. Clinical cases were the clearest facilitator because they matched the cognitive and production needs of multiple roles. The Art Guideline emerged because the project learned that clinically meaningful visual requirements were otherwise discovered too late. Intermediate validation loops were valuable because they moved defect detection closer to defect introduction. The main inhibitor was not a lack of process structure but the latency of expert feedback.

These results suggest three transferable propositions: boundary-artifact fit is role-dependent, not formality-dependent; visual assets in serious games can be safety- or learning-critical requirements belonging in the quality-assurance lifecycle; and stakeholder response time is itself a process dependency that methods should specify rather than assume.

Although PYPr was conceived for clinical serious games, several of its principles are independent of the healthcare domain. The study suggests that multidisciplinary game development benefits from treating domain artifacts as first-class development artifacts, introducing intermediate quality gates before implementation, and explicitly managing the translation of knowledge between specialists and software developers. These principles align with classical software engineering concerns such as requirements engineering, artifact coordination, lifecycle management, and quality assurance.

6.1 Relationship to Existing Approaches

To clarify what is inherited from Play Your Process versus what is original to PYPr: PYPr inherits from PYP the general premise that a structured domain process can ground game design and the use of a GDD as a project-level repository, adapting PYP's context-study and evaluation stages into domain-specific equivalents, replacing a single BPMN-based context study with three complementary medical artifacts and extending single-stage evaluation into two sequential validation gates; it introduces as original contributions the Art Guideline, the intermediate art-validation loop, and the explicit treatment of expert response time as a lifecycle dependency, none of which have an equivalent in PYP. Compared with process-oriented approaches that map a single formal model to game elements, PYP's BPMN-to-game mapping [5] and CTT-based task modeling [21], PYPr replaces a single-artifact mapping with a multi-artifact translation pipeline and adds explicit art- and game-validation loops; compared with requirements elicitation [6] and collaborative framework approaches [10, 12], which address isolated phases of development, PYPr provides a domain-specific handoff and validation structure spanning the full lifecycle.

The method should not be generalized prematurely: its current formulation targets a visual-novel architecture with minigames, and other genres (simulation, strategy, multiplayer, real-time 3D) raise different architecture, asset, and performance concerns. Nevertheless, the underlying process knowledge, domain-native boundary artifacts, early visual constraints, intermediate expert validation, and explicit dependency management are plausibly transferable across genres.

6.2 Threats to Validity

The principal threat to external validity is that PYPr was developed and evaluated within a single project, clinical domain, and institutional setting; the medical coordinators had participated in a previous game project, so team maturity and organizational continuity may partly explain positive outcomes, and replication with less experienced teams is necessary. A related threat concerns evaluator independence: most participants in the final evaluation were also members of the development team, and a single researcher embedded in the same project performed coding without independent double-coding, creating risks of researcher expectancy and confirmation bias. Triangulation across multiple data sources (meeting records, artifacts, direct observation) and the inclusion of one external participant partially mitigate this threat but do not eliminate it.

Construct validity is further limited by reliance on participant perception in a small sample of nine participants; claims about efficiency and rework reduction are therefore phrased as perceived or process-supported effects rather than objective productivity gains, and future evaluations should collect cycle time, blocked time, defect origin, rework, and validation latency metrics.

Finally, this study did not formally evaluate the game's learning effectiveness: the method evaluation concerns development support, communication, fidelity work, and process organization, and does not demonstrate that games produced with PYPr improve clinical learning or transfer to practice.

7 Conclusion and Future Work

This paper addressed a software engineering problem in serious-game development: translating safety-sensitive clinical knowledge into game design, visual assets, mechanics, and software through a multidisciplinary lifecycle. Through ADR, the project evolved from an unsuccessful direct transfer of a BPMN-centered method to Play Your Protocol, a process organized around domain-native medical artifacts, storyboards, an Art Guideline, explicit minigame design, and staged validation loops.

The evaluation indicates that clinical cases were the most consistently used bridge between medical knowledge and development decisions; shared artifacts and intermediate validation supported alignment, while delayed expert feedback remained the most important process bottleneck. PYPr is therefore not simply a game design method but an artifact-based software development process, and its contribution to software engineering for games is both a method and a set of process lessons: structure translation through artifacts that fit each role, treat clinically meaningful visual representation as a verifiable requirement, detect errors before they propagate across asset and code pipelines, and model expert response time as a lifecycle dependency.

Future work should replicate PYPr across other protocols, institutions, and team compositions, evaluate other game genres, instrument projects with objective process metrics, and add explicit feedback windows and escalation mechanisms; a further direction is selective tool support, where future tools could transform pathways and cases into traceable design elements and link validation findings to affected storyboards, assets, and code, rather than reinstating formal notation as a mandatory artifact. More broadly, this work argues that software engineering methods for serious games should explicitly manage how knowledge evolves across development artifacts: rather than relying solely on informal collaboration, multidisciplinary teams benefit from artifact-based communication, staged validation, and explicit quality requirements, and although evaluated in the healthcare domain, these principles may be applicable to other safety-critical serious games and multidisciplinary game development projects.

Artifact Availability

Supplementary material, including the PYPr evaluation questionnaire, the preliminary survey instrument, the rapid review protocols and search strings, and a description of the Art Guideline's structure, is available at https://github.com/LimaCarolineC/SE4Games2026.git.

Acknowledgements

The present work was supported by CAPES, the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) - Grant/Process number 409659/2025-8, and UFRJ - "Projetos Especiais do Parque Tecnológico".

AI Use Disclosure Statement

This article was created with support from generative AI, ChatGPT GPT-5.5 Thinking, used for English drafting from the original dissertation and structural revision. The authors remain responsible for the research claims, analysis, references, and final manuscript.

References

[1] A. Aster, C. Hütt, C. Morton, et al. 2024. Development and Evaluation of an Emergency Department Serious Game for Undergraduate Medical Students. *BMC Medical Education* 24 (2024), 1061. doi:10.1186/s12909-024-06056-z

[2] Laurence Bardin. 2011. *Análise de Conteúdo.* Almedina Brasil, São Paulo.

[3] T. M Classe, R. M. Araujo, and G. B. Xexéo. 2018. Process model game design: Uma ferramenta para apoio a sistematização de design de jogos digitais baseados em processos de negócio. *SBGAMES* (2018).

[4] T. M Classe, S. W. M. Siqueira, R. M. Araujo, and G. B. Xexéo. 2020. Play your process-um método de design de jogos digitais baseados em modelos de processos de negócio. *Anais Estendidos do XVI Simpósio Brasileiro de Sistemas de Informação (Anais Estendidos do SBSI 2020)* (2020), 142–157. doi:10.5753/sbsi.2020.13136

[5] Tadeu Moreira de Classe, Renata Mendes de Araujo, Geraldo Bonorino Xexéo, and Sean W. M. Siqueira. 2019. The Play Your Process Method for Business Process-Based Digital Game Design. *International Journal of Serious Games* 6, 1 (2019), 27–48.

[6] Oscar Colteli, Ximo Grandi, Ricardo Tosca, Pedro Latorre, José Salvador Sánchez, Luís Vicente Lizán, Francisco Ros-Bernal, and Conrrado Martínez-Cadenas. 2015. Designing Serious Games for Learning Support in Medicine Studies: A Specific Method to Elicit and Formalize Requirements. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*. IEEE, 1–4. doi:10.1109/FIE.2014.7044156

[7] Aisling Duffy, Neda Boroumandzad, Amanda L. Sherman, Gillian Christie, Imam Riadi, and Sylvain Moreno. 2025. Examining Challenges to Co-Design Digital Health Interventions With End Users: Systematic Review. *Journal of Medical Internet Research* 27 (2025), e50178. doi:10.2196/50178

[8] Marilyn J. Field and Kathleen N. Lohr. 1990. *Clinical Practice Guidelines: Directions for a New Program.* National Academy Press, Washington, DC.

[9] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. 2004. Design Science in Information Systems Research. *MIS Quarterly* 28, 1 (2004), 75–105. doi:10.2307/25148625

[10] David Jaccard, Laurent Suppan, Eric Sanchez, Alexandre Huguenin, and Michael Laurent. 2021. The co.LAB Generic Framework for Collaborative Design of Serious Games: Development Study. *JMIR Serious Games* 9, 3 (2021), e28674. doi:10.2196/28674

[11] Jason Jordan, Jennifer Wagner, David E. Manthey, Markus Wolff, Sally Santen, and Sarah J. Cico. 2019. Optimizing Lectures From a Cognitive Load Perspective. *AEM Education and Training* 4, 3 (2019), 306–312. doi:10.1002/aet2.10389

[12] Janne M. Koivisto, Elina Haavisto, Hanna Niemi, Pekka Haho, Susanna Nylund, and Jari Multisilta. 2018. Design Principles for Simulation Games for Learning Clinical Reasoning: A Design-Based Research Approach. *Nurse Education Today* 60 (2018), 114–120. doi:10.1016/j.nedt.2017.10.002

[13] Gesa Krause-Jüttler, Jan Weitz, and Ulrich Bork. 2022. Interdisciplinary Collaborations in Digital Health Research: Mixed Methods Case Study. *JMIR Human Factors* 9, 2 (2022), e36579. doi:10.2196/36579

[14] Jadete Barbosa Lampert, Nilce Maria da Silva Campos Costa, Gianna Lepre Perim, Ively Guimarães Abdalla, Rinaldo Henrique Aguilar-da Silva, and Regina Celes de Rosa Stella. 2009. Tendências de mudanças em um grupo de escolas médicas brasileiras. *Revista Brasileira de Educação Médica* 33, 1 Supl. 1 (2009), 19–34.

[15] Caroline Lima, Geraldo Xexéo, and Tadeu Classe. 2025. Beyond Play Your Process - Game Design and Business Process Model: A Rapid Review. In *Anais do XXIV Simpósio Brasileiro de Jogos e Entretenimento Digital* (Salvador/BA). SBC, Porto Alegre, RS, Brasil, 35–47. doi:10.5753/sbgames.2025.9632

[16] Heidi L. Lujan and Stephen E. DiCarlo. 2025. The Paradox of Knowledge: Why Medical Students Know More But Understand Less. *Medical Science Educator* 35, 3 (2025), 1761–1766. doi:10.1007/s40670-025-02379-8

[17] Aisling M. O'Carroll, Erin P. Westby, John Dooley, and Kevin E. Gordon. 2015. Information-Seeking Behaviors of Medical Students: A Cross-Sectional Web-Based Survey. *JMIR Medical Education* 1, 1 (2015), e4. doi:10.2196/mededu.4267 Published online: 29 Jun 2015.

[18] Mor Peleg. 2008. Computer-Interpretable Clinical Guidelines: A Methodological Review. *Journal of Biomedical Informatics* 41, 4 (2008), 744–763.

[19] Maung K. Sein, Ola Henfridsson, Sandeep Purao, Matti Rossi, and Rikard Lindgren. 2011. Action Design Research. *MIS Quarterly* 35, 1 (2011), 37–56.

[20] Sissel Steihaug, Anne K. Johannessen, Marian Ådnanes, Børge Paulsen, and Russell Mannion. 2016. Challenges in Achieving Collaboration in Clinical Practice: The Case of Norwegian Health Care. *International Journal of Integrated Care* 16, 3 (2016), 3. doi:10.5334/ijic.2217

[21] Alberto Cabas Vidani and Luca Chittaro. 2009. Using a Task Modeling Formalism in the Design of Serious Games for Emergency Medical Procedures. In *2009 Conference in Games and Virtual Worlds for Serious Applications*. IEEE, 95–102. doi:10.1109/VS-GAMES.2009.24

[22] James Q. Young, Jeroen J. G. Van Merrienboer, Steven Durning, and Olle Ten Cate. 2014. Cognitive Load Theory: Implications for Medical Education: AMEE Guide No. 86. *Medical Teacher* 36, 5 (2014), 371–384. doi:10.3109/0142159X.2014.889290