Mr. OCL: An Escape Room Board Game for Teaching Object Constraints

Victor Travassos Sarinho
Universidade Estadual de Feira de Santana
Feira de Santana, Brasil
vsarinho@uefs.br

Abstract

The Object Constraint Language (OCL) plays a fundamental role in Model-Driven Software Engineering by enabling the formal specification of invariants, preconditions, and postconditions over UML models. However, its declarative nature and high level of abstraction make OCL difficult to teach and learn. Although game-based learning has been successfully applied to several Software Engineering topics, educational games specifically designed to support collaborative learning of OCL remain scarce. This paper presents **Mr. OCL**, an escape room-inspired board game that teaches formal object constraints through progressively structured challenges based on UML class and object diagrams. Players collaboratively solve each challenge by constructing sequences of action cards governed by OCL contracts, while a Game Master validates their logical consistency. A preliminary classroom evaluation based on the MEEGA+ model revealed positive student perceptions regarding engagement, collaboration, and perceived learning, suggesting that Mr. OCL is a promising approach that warrants further investigation through controlled studies of OCL learning outcomes.

Keywords

Object Constraint Language, Gamification, Software Engineering Education, Escape Room, Board Games

1 Introduction

The increasing adoption of Model-Driven Software Engineering (MDSE) has reinforced the importance of formal specification languages capable of complementing UML models with precise semantic constraints. Among these languages, the Object Constraint Language (OCL) has become the de facto standard for expressing invariants, preconditions, postconditions, and queries on UML models, enabling the specification of behavioral and structural properties that cannot be adequately represented by diagrams alone [1, 2]. Consequently, OCL plays a fundamental role in software verification, model validation, model transformation, and code generation.

Despite its practical relevance, teaching OCL remains challenging in Software Engineering Education. Its declarative nature requires students to reason about system states, object navigation, collection operators, and logical expressions simultaneously, demanding a level of abstraction that is considerably higher than that required for conventional UML modeling. As a result, students often experience difficulties relating formal constraints to the behavior of software models, reducing both motivation and learning effectiveness [2].

Several educational strategies have been proposed to address these challenges, including educational games, collaborative activities, and interactive modeling environments. Although these approaches have produced encouraging results in different Software Engineering topics, the teaching of formal object constraints remains comparatively underexplored, particularly regarding collaborative and tangible learning experiences. Existing educational games primarily focus on UML diagram construction, programming, or software development practices, leaving OCL with limited pedagogical support.

Motivated by this gap, this paper presents **Mr. OCL**, an escape room-inspired board game designed to support the teaching of OCL through collaborative logical reasoning. The game comprises four sequential challenges that progressively introduce core OCL concepts, including attribute manipulation, collection operations, numerical invariants, and first-order logic. By combining UML models, OCL contracts, tangible board game mechanics, and human-mediated validation, the proposed approach seeks to make formal specification concepts more accessible through collaborative and active learning.

2 Theoretical Background

Game-Based Learning (GBL) has become an established pedagogical approach in Computing Education by promoting active learning through problem solving, experimentation, collaboration, and immediate feedback. Rather than replacing traditional teaching methods, educational games complement classroom instruction by creating learning environments in which students actively construct knowledge while remaining engaged throughout the learning process. Likewise, gamification incorporates game elements into non-game educational activities to increase motivation and participation, particularly when game mechanics are aligned with well-defined learning objectives [3, 7].

Among GBL strategies, educational escape rooms have received increasing attention in STEM education because they naturally integrate collaborative problem solving with progressively structured challenges. Players must collectively solve interconnected puzzles whose solutions unlock subsequent stages, encouraging communication, critical thinking, and iterative refinement of hypotheses. Recent systematic reviews have shown positive effects on student engagement, teamwork, and knowledge acquisition, indicating that escape rooms constitute an effective learning strategy for educational scenarios involving logical reasoning and sequential decision making [5].

The use of educational games in Software Engineering Education has also expanded considerably over the last decade. Existing approaches address topics such as introductory programming, requirements engineering, software testing, project management, and conceptual modeling. Within conceptual modeling education, several gamified environments have been proposed to support UML learning through quizzes, role-playing activities, modeling challenges, and automatic feedback mechanisms. These approaches have demonstrated improvements in students motivation and understanding of UML notation, although their primary emphasis remains on diagram construction rather than formal specification [4].

An analysis of these educational approaches reveals substantial differences in their learning objectives and interaction mechanisms. UML-oriented educational games primarily emphasize diagram construction and notation, whereas educational escape rooms emphasize collaborative problem solving, sequential challenges, and engagement. In contrast, OCL-oriented learning resources tend to focus on the interpretation and specification of formal constraints, often relying on digital modeling environments or conventional exercises. These approaches address complementary aspects of Software Engineering Education, but they leave relatively unexplored the combination of formal constraint reasoning, collaborative physical interaction, and progressive escape-room challenges.

Mr. OCL addresses this intersection by treating OCL contracts not merely as textual expressions to be written or memorized, but as rules that govern tangible state transitions. Action cards expose preconditions and postconditions, while UML diagrams provide the system state that students must reason about. This design therefore differs from UML gamification approaches centered on diagram construction [4] and from escape-room approaches whose primary emphasis is collaborative problem solving rather than formal specification [5]. As a result, Mr. OCL intends to make students reason about constraint satisfaction and state transitions collaboratively, while progressively introducing increasingly complex OCL constructs.

3 Methodology

The methodology of *Mr. OCL* is rooted in *Scaffolded Learning* and *Analog Game-Based Learning*. The design transitions from a digital-dependent validation to a human-mediated logical verification, where the instructor or a designated "Game Master" acts as the formal reasoning engine.

The game is structured into four sequential rooms, each isolating specific OCL constructs through progressively complex logical puzzles, allowing students to build on previously introduced concepts before encountering more complex logical expressions: **Room 1 (The Gatekeeper)**, focused on basic attributes and state comparisons; **Room 2 (The Archive)**, introducing collection membership (`includes`, `excludes`) and association navigation; **Room 3 (The Engine Room)**, exploring arithmetic operations and numerical invariants (e.g., `sum`, `size`); and **Room 4 (The Inner Sanctum)**, covering first-order logic, including universal quantification (`forAll`) and implications (`implies`). This ordering was designed to progressively increase the scope of reasoning, moving from individual object states to relationships among objects, aggregate properties over collections, and finally system-wide quantified and logical constraints.

As a physical board game, *Mr. OCL* utilizes tangible components to represent abstract models:

- **Action Cards:** Each card represents an OCL operation. The cards explicitly display the *pre-conditions* (requirements to play) and *post-conditions* (the resulting state change).
- **The Board:** Displays an UML Class Diagram and a mutable Object Diagram where players visualize the current state of the current played "room".

Players must collaboratively negotiate the sequence of cards, ensuring that the post-condition of a preceding card satisfies the pre-condition of the subsequent one.

In this analog version, the validation of OCL constraints according to selected action cards is performed by a **Game Master (GM)**, which is typically the instructor or a student with the necessary knowledge of the room's logic. Thus, for the Human-Mediated Validation process, the following steps are performed:

1. **Sequence Submission:** Players present a full sequence of action cards intended to solve the room's challenge.
2. **The "Logical Oracle":** The GM, holding the "Master Solution" (previously verified during the game's design phase), validates the sequence.
3. **Feedback and Constraints:** The GM provides immediate feedback on which specific pre-condition or invariant was violated. To increase tension and discourage "trial and error", the GM also manages a limited number of attempts, forcing students to verify their logic manually before submission.

The absence of an automated digital interface encourages deeper peer-to-peer discussion. Students are forced to "trace" the OCL expressions mentally or on paper, simulating the manual inspection process often required in professional code reviews and formal modeling. The game thus transforms OCL from a syntax-heavy exercise into a strategic negotiation of system states.

Finally, the UML Specification Environment (USE)[1] tool was adopted as the formal modeling and validation environment supporting the game design process. In addition to generating UML class and object diagrams, USE provides an executable environment in which commands can modify object attributes, states, and associations while automatically verifying the satisfaction of OCL invariants, preconditions, and postconditions. Consequently, each room was specified as a UML model enriched with OCL constraints, and every action card was derived from one or more executable commands defined on these models. By simulating complete sequences of action cards through the corresponding command executions in USE, it was possible to verify the correctness of the proposed solutions, identify invalid state transitions and contradictory contracts, and generate the verified master solutions used by the Game Master during gameplay.

4 Game Design and Implementation

The development of *Mr. OCL* resulted in a set of physical and conceptual artifacts designed to facilitate the understanding of formal constraints. This section describes the game's narrative phases, the underlying UML models, and the action card system.

4.1 The Narrative Arc: The Rescue of Logic

The game is set inside the *Logic Core*, a fictional digital infrastructure responsible for maintaining the consistency of software models. This system has been corrupted by the *Null Pointer*, a malicious entity that disrupts logical consistency by violating contracts and

[1] https://github.com/useocl/use

destabilizing object states. Players assume the role of *Mr. OCL*, an elite software agent specialized in restoring the integrity of model-driven systems through formal reasoning. Rather than defeating enemies through combat, Mr. OCL overcomes each challenge by restoring valid system states according to OCL contracts.

The adventure is divided into four sequential rooms, each representing a different subsystem of the Logic Core. Every room presents an initial corrupted object diagram together with a UML class model and a set of action cards describing operations through preconditions and postconditions. Players escape each room only by constructing a valid sequence of actions that transforms the system from its initial configuration into a state satisfying all invariants and mission objectives.

4.2 Room 1: Attribute-Based Constraints

Narrative and Context: The journey begins at the "Authentication Gate" of the Logic Core. The Null Pointer has corrupted the authentication protocols, the active user is unregistered, and the access card has expired. Mr. OCL must restore authentication before entering the system.

Room Mission: The user (`u1.User`) must obtain the required authorization (`u1.hasKey = true`).

Technical Challenge: This initial phase focuses on **attribute-based pre-conditions**. Students must synchronize user Credentials with Terminal states by reasoning about simple boolean and enumeration attributes. The class and object diagrams representing this scenario are shown in Figure 1.

Action Cards and Logic: The 8 available action cards for this room (Figure 2) represent operations such as activating an access card, authorizing a user, and validating an entry request. Each operation explicitly specifies the conditions that must hold before execution, together with the resulting updates to the object diagram.

The Escape Sequence: To successfully "escape" this room, players must identify a strict dependency chain. The required sequence of operations is:

(1) **Initialize Card (`activateCard`):** This operation must be performed first to transition the card state from `EXPIRED` to `ACTIVE`.
(2) **Register User (`authorizeUser`):** Since the card is now `ACTIVE`, the pre-condition for authorization is satisfied, allowing the user to be added to the allowed list.
(3) **Final Scan (`attemptEntry`):** The final step to reach the winning state.

Pedagogical Goal: The complexity of this room lies in the **state dependency** within OCL contracts. If a player attempts to *Register User* before *Initialize Card*, the Game Master will reject the move because the `pre: card.status = #Active` condition is not met. This enforces a rigorous understanding of how sequential actions in a system are constrained by formal logic.

4.3 Room 2: Collection and Association Constraints

Narrative and Context: Mr. OCL enters the "Archive of Forgotten Scripts". The *Null Pointer* has scattered sensitive data across unauthorized directories. Although the system's invariants remain intact, the "Security Clearance" protocol is inactive. Players must

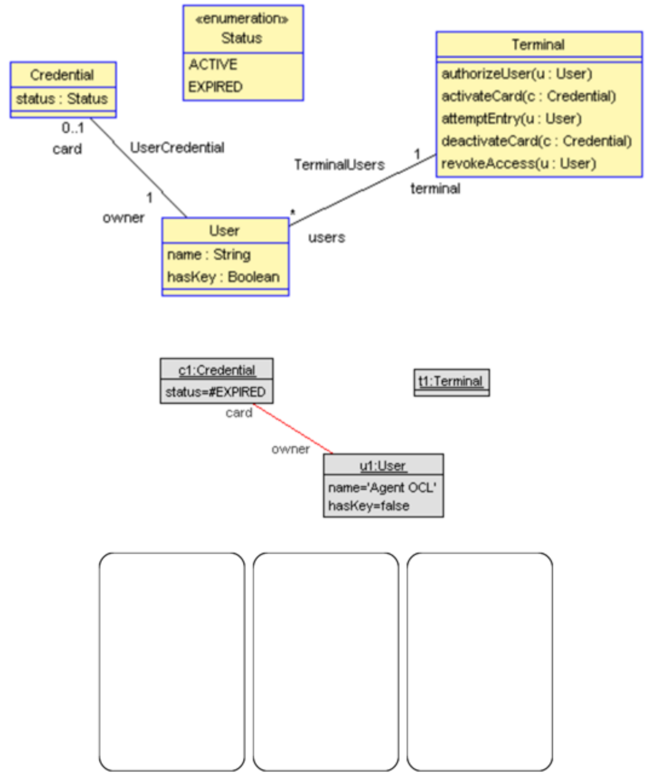

Figure 1: UML Diagrams and expected action cards response of Room 1.

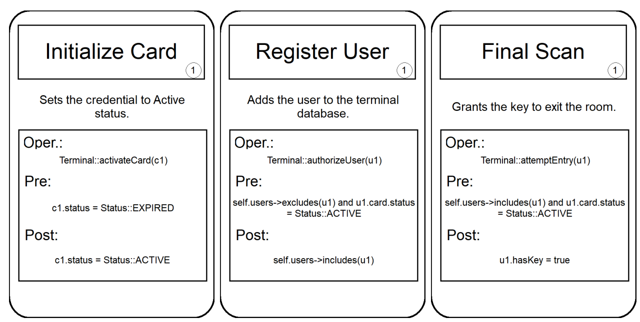

Figure 2: Examples of action cards of Room 1.

move all sensitive documents to an encrypted folder and populate the access list to restore system integrity.

Room Mission: The user (`u1.User`) must successfully download the decrypted scripts (`u1.hasBlueprints = true`).

Technical Challenge: This phase introduces **navigation by associations** and **collection operations**. Students must move beyond simple attribute checks to reason about relationships between objects, specifically focusing on one-to-many and many-to-many associations. The pedagogical focus is on OCL collection operators such as `includes()`, `excludes()`, and `size()`. The UML models for this room are depicted in Figure 3.

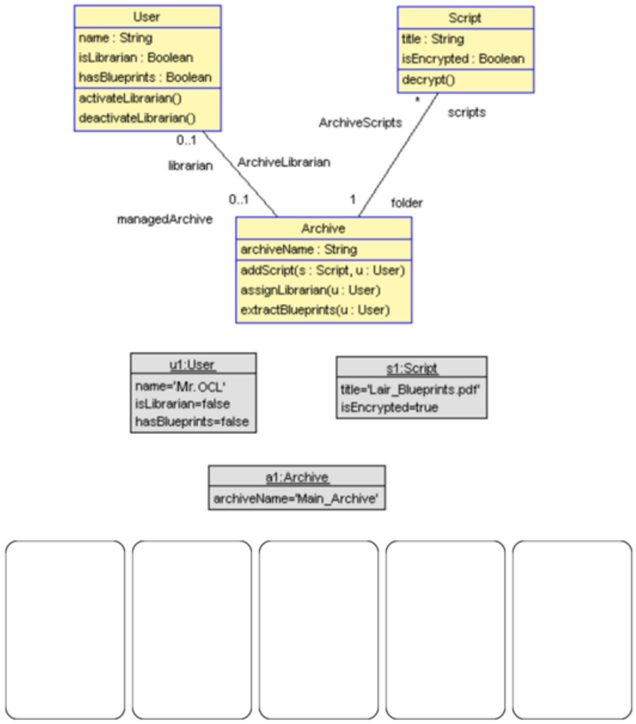

Figure 3: UML Diagrams and expected action cards response of Room 2.

Action Cards and Set Logic: The 9 available cards for this room (Figure 4) represent operations on collections and state transitions within the *Archive*, as defined in the provided USE model. The **Index File `addScript`** card focuses on collection integrity, requiring that a script is not already present (`excludes`) and is not encrypted before being added. The **Login Terminal `assignLibrarian`** card manages the association between a `User` and the `Archive`, requiring the librarian role to be active. Finally, the **Download Data `extractBlueprints`** card represents a complex contract that navigates the association to verify both the assigned librarian and the non-empty state of the script collection.

The Escape Sequence: The solution requires the player to manage the user's operational role through a precise sequence of state changes:

(1) **New Librarian (`activateLibrarian`) card:** The player must first elevate the "Mr. OCL" status to `isLibrarian = true`. This act of "logging in" is the prerequisite for all subsequent administrative operations in the room.
(2) **Login Terminal (`assignLibrarian`) card:** The user is formally linked to the `Main_Archive`. This step solidifies the responsibility association required for the final extraction.
(3) **Decryption Tool (`decrypt`) card:** The player targets the "Lair_Blueprints.pdf" file to flip its state. Since `addScript` requires a decrypted file, this transformation is essential.
(4) **Index File (`addScript`) card:** As "Mr. OCL" is currently in the librarian role, the decrypted script is successfully added to the archive's collection.

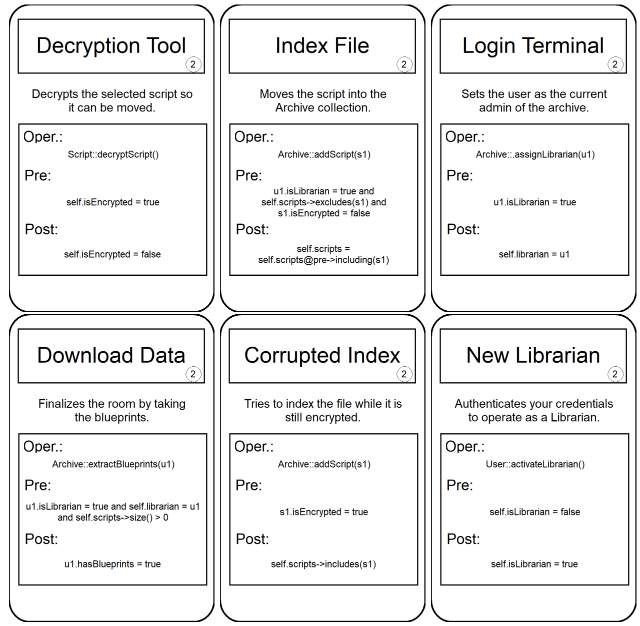

Figure 4: Examples of action cards of Room 2.

(5) **Download Data (`extractBlueprints`) card:** The victory condition is met only if the user remains "logged in" as a librarian, is correctly associated with the archive, and the script collection is populated.

Pedagogical Goal: Room 2 shifts the focus from simple state-checking to the *interdependency of object properties and associations*. By requiring a sequence that involves both collection operators (like `excludes` and `size`) and association management (the `librarian` role), students learn how OCL contracts enforce structural integrity across different entities in a system.

4.4 Room 3: Arithmetic and Aggregate Constraints

Narrative and Context: *Mr. OCL* reaches the "Engine Room", responsible for distributing power throughout the Logic Core. The *Null Pointer* has overloaded some Energy Nodes while leaving others underutilized. If the total energy exceeds the safety threshold, the Logic Core will shut down. Players must rebalance the grid while preserving all numerical constraints.

Room Mission: The user (`u1.User`) must obtain the authorization key (`u1.hasAuthorizedKey = true`) by successfully balancing the system's energy.

Technical Challenge: This room explores **arithmetic aggregates** and **numeric invariants**. Students are required to use OCL operations that perform calculations over collections, specifically `sum()` and `size()`, combined with relational operators (>, <, =). The focus shifts to ensuring that the sum of a specific attribute across multiple objects remains within a valid range. The UML models representing this room are illustrated in Figure 5.

Action Cards and Aggregation Logic: The 9 available cards for this room (Figure 6) manipulate the `DistributionHub` while

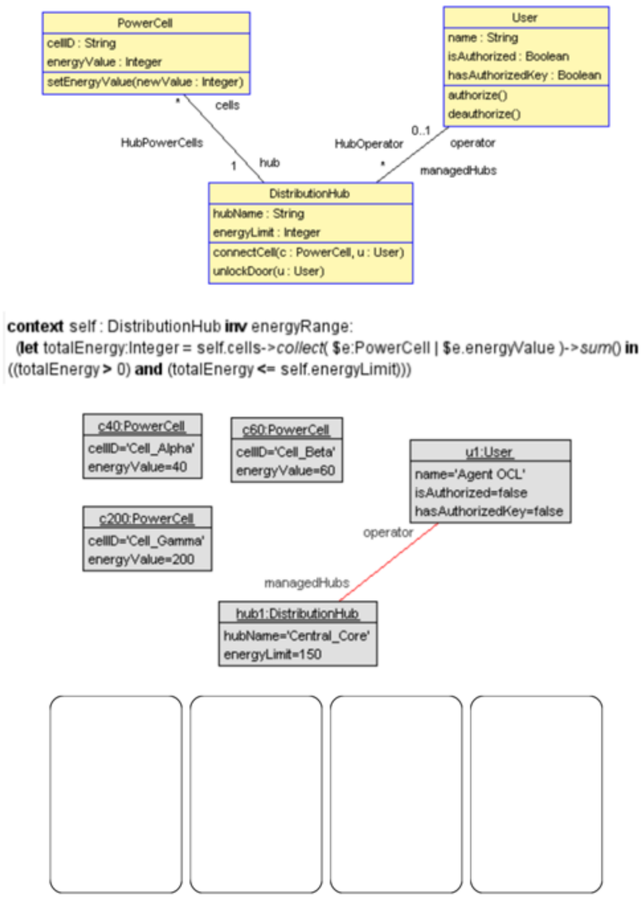

Figure 5: UML Diagrams and expected action cards response of Room 3.

preserving the subsystem's numerical consistency. The *New Hub Oper* card authorizes the player to perform administrative operations, enabling the connection of new `PowerCell` instances. The *Plug Alpha* and *Plug Beta* cards add energy cells to the network, requiring players to evaluate how each contribution affects the overall energy balance. Once the required equilibrium is achieved, the *Terminal Override* card unlocks the exit by validating the aggregate energy of the connected cells. The deck also includes distractor cards that introduce excessive energy, infeasible preconditions, or invalid state transitions, encouraging students to identify violations of numerical constraints and global invariants before executing an operation.

The Escape Sequence: To successfully escape this room, players must identify a sequence that satisfies both the authorization chain and the numerical equilibrium constraint:

(1) **New Hub Oper (`authorize`):** Must be performed first to enable all subsequent `connectCell` operations. Without authorization, the `hubOperator` association is not established.
(2) **Plug Alpha (`connectCell c40`):** Connects the first cell. The energy sum becomes 40. *Plug Alpha* and *Plug Beta* can alternate positions, but both must be played before *Terminal Override* card.
(3) **Plug Beta (`connectCell c60`):** Connects the second cell. The energy sum becomes exactly 100, which satisfies the *Terminal Override* pre-condition.
(4) **Terminal Override (`unlockDoor`):** The final step. The precondition `self.cells.energyValue->sum() = 100` is satisfied, granting `u1.hasAuthorizedKey = true` and completing the room.

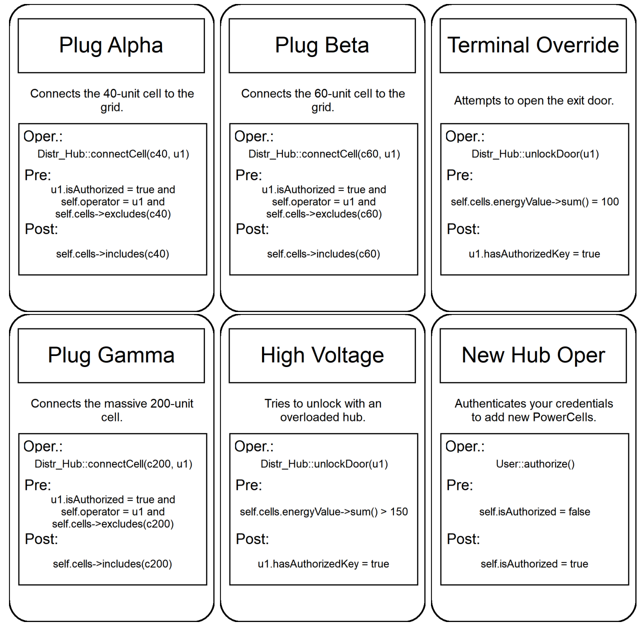

Figure 6: Examples of action cards of Room 3.

Pedagogical Goal: Room 3 teaches that OCL is not just for checking "true/false" states, but also for *mathematical integrity*. By calculating sums across associations, students learn how local object changes propagate to affect global system constraints.

4.5 Room 4: Quantified and Logical Constraints

Narrative and Context: Mr. OCL reaches the "Inner Sanctum", the final defense layer. The *Null Pointer* hides behind a shield powered by `SecurityNode` instances. There is no single object to fix, but a entire network of interconnected nodes. The room is in a "Deadlock" state, where the shield deactivates only when a specific logical configuration is met across all active nodes simultaneously.

Room Mission: The user (`u1.User`) must rescue the Logic Core (`u1.hasRescuedCore = true`).

Technical Challenge: This phase introduces **first-order logic** and **complex quantifiers**. Students must master universal quantification (`forAll`), existential quantification (`exists`), and logical implications (`implies`). The challenge shifts from local or aggregate arithmetic to verifying global properties that must hold true for entire sets of objects. The UML models representing this room are illustrated in Figure 7.

Action Cards and Advanced Logic: The 9 available cards for this room (Figure 8) represent operations that progressively restore

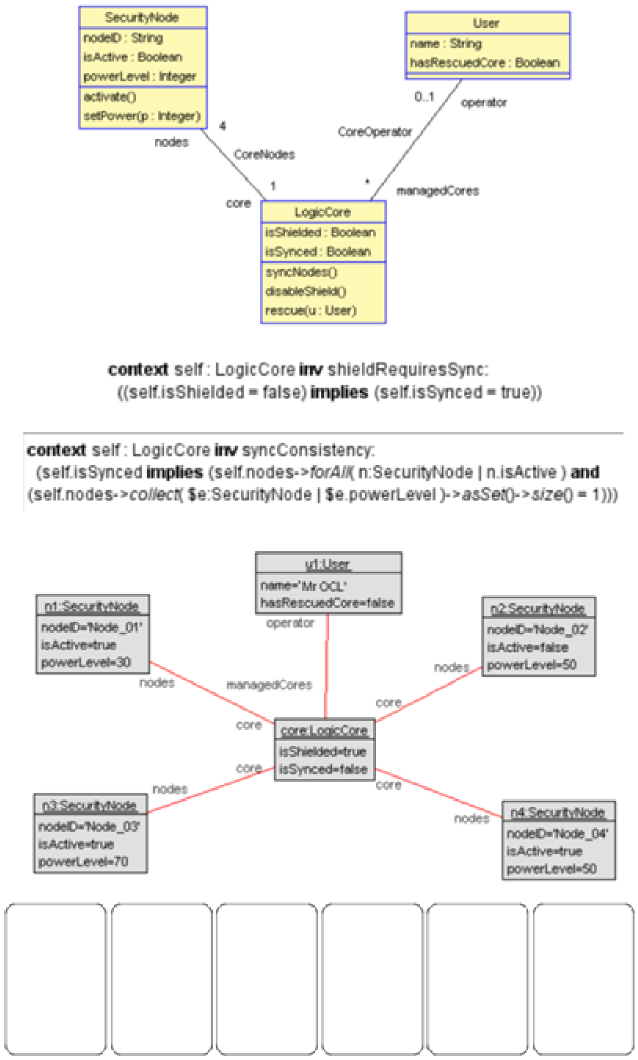

Figure 7: UML Diagrams and expected action cards response of Room 4.

the logical consistency of the security network. The *Reboot* and *Calibrate* cards modify the activation state and power level of individual `Security Node` instancess, preparing the system to satisfy the global constraints. The *Logic Sync* card represents the central challenge of the room, requiring that all nodes are active and share the same power level before synchronization can occur. Once the network has been synchronized, the *Shield Down* card disables the protective barrier by validating both the synchronization state and the overall energy distribution. Finally, *Final Extraction* completes the mission by rescuing the Logic Core. The deck also includes distractor cards containing contradictory or incomplete contracts, encouraging students to identify invalid quantified expressions and logical dependencies before executing an operation.

The Escape Sequence: The solution requires a carefully orchestrated sequence that progressively satisfies nested quantifiers and implications:

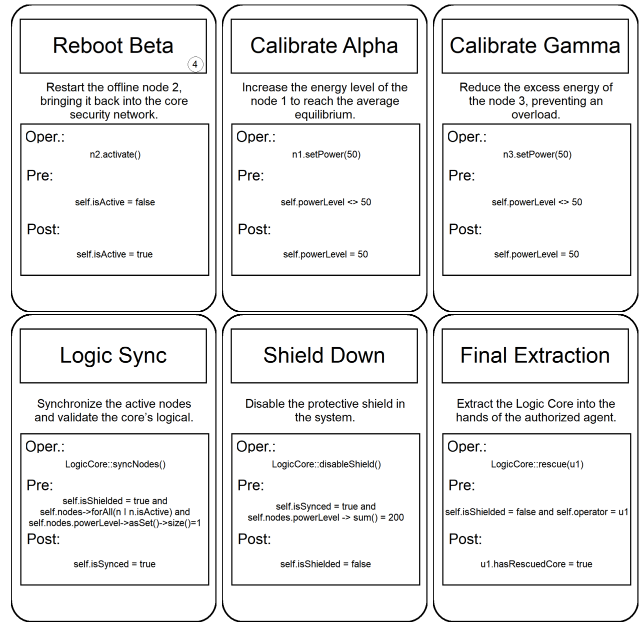

Figure 8: Examples of action cards of Room 4.

(1) **Reboot Beta (`activate`):** Must be performed first to bring n2 online. Without this step, the `forAll(n | n.isActive)` condition in *Logic Sync* can never be satisfied.
(2) **Calibrate Alpha (`setPower 50`):** Adjusts n1 from 30 to 50. The order among the two *Calibrate* cards is flexible, but all must be completed before *Logic Sync*.
(3) **Calibrate Gamma (`setPower 50`):** Adjusts n3 from 70 to 50.
(4) **Logic Sync (`syncNodes`):** The turning point. With all nodes active and uniformly calibrated, the pre-conditions are met, setting `isSynced = true`.
(5) **Shield Down (`disableShield`):** Now that `isSynced = true` and the total power is 200, the shield can be safely disabled (`isShielded = false`), satisfying the `shieldRequiresSync` invariant.
(6) **Final Extraction (`rescue`):** With the shield down and u1 as the operator, the core is rescued (`u1.hasRescuedCore = true`), completing the game.

Pedagogical Goal: Room 4 concludes the game's pedagogical progression and serves as the ultimate test of students' logical mindset. Integrating attributes, collections, numerical aggregates, and first-order logic, players construct valid action sequences using `forAll`, `exists`, and `implies` to satisfy system-wide constraints. Finally, the room shows that OCL is a cohesive language for expressing global system properties, promoting semantic reasoning about formal specifications rather than syntax memorization.

5 Game Evaluation

The evaluation was conducted as part of a Software Engineering course in an undergraduate Computer Engineering program (Fig. 9). The game was applied as a complementary activity following a practical exercise with the USE tool, which ensured that participants

Figure 9: Undergraduate students participating in an experimental session of Mr. OCL within a classroom environment.

already possessed prior knowledge of OCL. Participants were organized into distinct groups of 4 students, and the game session lasted approximately 60-80 minutes. A Game Master, acting as an instructor, controlled the sequence of challenges, provided feedback during gameplay, and validated submitted action sequences according to the previously verified master solutions. At the end of the game session with 32 students, 13 participants answered a questionnaire composed of open-ended questions and items based on the MEEGA+ [6], using a 5-point Likert scale ranging from $[-2, 2]$.

Regarding the quantitative analysis results, **Social Interaction** was the most robust dimension (median = 2 across all items). "*I was able to interact with other people*" achieved unanimous positive response (10 Strongly Agree, 3 Agree), confirming collaboration as the most successful pillar. **Fun** also performed strongly (median = 2 for "*had fun*"). **Learnability** presented high scores: "*learning was easy*" and "*most people learn quickly*" achieved medians of 2 (8 and 7 Strongly Agree, respectively). However, "*few things to learn before playing*" received median = 1, indicating a moderate initial learning curve. **Operability** was well evaluated: "*game easy to play*" (median = 1) and "*rules are clear*" (median = 2, 9 Strongly Agree). **Accessibility** also performed strongly: "*fonts readable*" achieved median = 2 (9 Strongly Agree). By contrast, **Aesthetics** was the weakest dimension: "*design is attractive*", "*texts/colors/fonts consistent*", and "*colors clear*" all received medians of 1 (4–5 Indifferent), indicating visual appeal did not meet expectations. **Focused Attention** showed mixed results: "*interesting beginning*" (median = 1), but "*lost track of time*" obtained median = 0 (4 Indifferent, 3 Disagree), suggesting moderate immersion. **Confidence** and **Satisfaction** obtained medians of 1, except "*sense of accomplishment*" (median = 2). **Relevance** and **Perceived Learning** demonstrated strong results: "*clear relation to course*", "*appropriate teaching method*", and "*contributed to learning*" achieved medians of 2 (8–9 Strongly Agree). However, "*effective compared to other activities*" and "*contributed to OCL concepts*" received medians of 1, indicating that students did not yet fully perceive the game as superior to traditional instructional methods or as effectively supporting OCL concept acquisition.

Challenge presented dispersion: "*appropriate pace*" (median = 2), but "*does not become monotonous*" (median = 1, 5 Indifferent).

The qualitative data also corroborate the quantitative findings. Most valued aspects were the Game Master dynamic ("*the dynamic between GM and players makes the game more fluid*"), narrative immersion ("*reminds me of an RPG*"), cooperative gameplay ("*facilitates communication*"), and intellectual challenge ("*challenging and balanced*"). Areas for improvement converged on four themes: (i) **more content**, "*more room scenarios and cards*"; (ii) **Game Master role**, "*master could participate more actively*" and "*summarize the rules*"; (iii) **clarity**, "*ambiguous what the room needs and how it relates to classes and cards*"; and (iv) **environment**, "*problems with physical setting*".

6 Conclusions and Future Work

Teaching OCL remains a significant challenge, largely due to the gap between abstract logical reasoning and its practical application in software modeling. Mr. OCL addresses this challenge by introducing a collaborative, escape room–inspired board game that encourages students to actively engage with UML models and object constraints through problem solving and structured reasoning.

The proposed game abstracts the complexities of OCL into a tangible Action Card mechanic, enabling students to focus on the interpretation and application of invariants, preconditions, and postconditions rather than tool-specific syntax. Through a progressive sequence of challenges, players advance from reasoning about object states and associations to aggregate expressions and first-order logic, while the Game Master provides human-mediated validation that preserves the rigor of formal specifications without requiring specialized software during gameplay.

The preliminary MEEGA+ evaluation indicates positive student perceptions regarding Social Interaction, Learnability, and Perceived Learning, suggesting that the game can support collaborative engagement with formal OCL concepts. Qualitative feedback further reinforced these findings by highlighting the value of the Game Master mechanism, cooperative problem solving, and the logical reasoning required to solve the challenges. However, these findings should not be interpreted as evidence of learning gains, since the study did not include pre- and post-tests or a comparison group. At the same time, the evaluation identified opportunities for improvement in the game's visual presentation, narrative immersion, and the expansion of gameplay content.

Future work includes conducting controlled empirical studies with larger and more diverse student populations to further investigate the impact of Mr. OCL on OCL learning, particularly regarding knowledge retention and students' ability to independently formulate constraints after the intervention. In addition, the game will be extended with new rooms covering more advanced OCL constructs, while its mechanics, narrative, and visual design will be refined based on the feedback obtained during the evaluation.

Artifact Availability

The game materials, including action cards, board layouts, UML models, USE specifications and master solutions are publicly available at https://drive.google.com/drive/folders/1ZjLyTR0jHZsWwrjj-hx_UUQCNUbVl-2n?usp=sharing.

References

[1] Marco Brambilla, Jordi Cabot, and Manuel Wimmer. 2017. *Model-Driven Software Engineering in Practice* (2nd ed.). Morgan & Claypool Publishers.

[2] Jordi Cabot and Martin Gogolla. 2012. Object constraint language (OCL): A Definitive Guide. In *International school on formal methods for the design of computer, communication and software systems*. Springer, 58–90.

[3] Thomas M Connolly, Elizabeth A Boyle, Ewan MacArthur, Thomas Hainey, and James M Boyle. 2012. A systematic literature review of empirical evidence on computer games and serious games. *Computers & education* 59, 2 (2012), 661–686.

[4] Giacomo Garaccione, Riccardo Coppola, Luca Ardito, and Marco Torchiano. 2026. Gamification of Conceptual Modeling Education with UML Class Diagrams: An Experimental Analysis. *Software and Systems Modeling* 25, 1 (2026), 239–270.

[5] Melissa Gruber and Stefanie Faßbender. 2025. Digital Educational Escape Games in STEM Higher Education: A Systematic Literature Review. *New Directions in the Teaching of Natural Sciences* 20 (2025).

[6] Giani Petri, Christiane Gresse Von Wangenheim, and Adriano Ferreti Borgatto. 2019. MEEGA+: Um Modelo para a Avaliação de Jogos Educacionais para o ensino de Computação. *Revista Brasileira de Informática na Educação* 27, 03 (2019), 52–81.

[7] Jan L Plass, Bruce D Homer, and Charles K Kinzer. 2015. Foundations of game-based learning. *Educational psychologist* 50, 4 (2015), 258–283.