

A Tycoon-Based Serious Game Proposal for Learning Computer Architecture

Arthur Gonçalves¹, Fabio Augusto Menocci Cappabianco¹

¹Instituto de Ciência e Tecnologia – Universidade Federal de São Paulo (UNIFESP)
Caixa Postal 12247-014 – São José dos Campos – SP – Brasil

Abstract. Introduction: *Computer Architecture and Organization is essential in computing education, yet students often struggle to relate digital components, processor organization, pipelining, and parallel execution. Objective:* *This paper presents an ongoing serious game that uses tycoon and idle mechanics to support learning in computer architecture and digital circuits. Methodology or Steps:* *The Unity prototype combines historical progression, from vacuum tubes to modern architectures, with a mining/combat loop where processor mechanisms affect performance. Expected Results:* *The project aims to improve engagement by turning abstract architecture concepts into visible gameplay rules. Future evaluation will use questionnaires and gameplay logs. Keywords* *serious games, computer architecture education, educational games, tycoon games, gamification.*

1. Introduction

Computer Architecture and Organization (CAO) connects digital logic, processor organization, memory hierarchy, instruction execution, and performance mechanisms (Null 2023). These concepts are fundamental in computing curricula, yet they are usually introduced through static diagrams, formal descriptions, and isolated simulators. As a result, students often understand terms such as instruction pipeline, multicore execution, or dynamic scheduling in a declarative way, without practical intuition about their effect on performance and system behavior. This gap between declarative knowledge and operational intuition is a recurring difficulty in CAO education, where the distance between physical components and higher-level abstractions is well known (Larraz-Mendiluze et al. 2024). Common mitigations—instruction-set simulators, datapath visualization, and game-based approaches—remain mostly simulator-centric and rarely sustain long-term, playful engagement with the underlying mechanisms.

Serious games support learning when they connect explicit educational goals with meaningful actions, feedback, and challenge (Dörner et al. 2016; Ma and Oikonomou 2017), and have been used in computing education to motivate practice and visualize abstract processes (de Camargo et al. 2025). Building on this, we propose *BitCore Tycoon*, a serious game concept and prototype in which CAO concepts become progression mechanics within a tycoon/idle structure, organized around the historical evolution of computing components and a second loop inspired by cryptocurrency mining. Instead of explanatory text, the game assigns mechanical roles to architectural concepts: components increase production, pipeline-like mechanisms improve repeated actions, and multicore systems allow simultaneous processing. Following *procedural rhetoric* (Bogost 2007), its argument is expressed through rules:

the player learns *why* a mechanism matters by experiencing how it changes production, speed, and throughput. The contribution is threefold: (i) an integrated educational gameplay loop connecting historical progression, architecture mechanisms, and a simulated economy; (ii) its framing as a Design Science Research artifact with an explicit concept-to-mechanic mapping; and (iii) a preliminary evaluation of the first environment. A glossary additionally lets curious players explore historical details and functional explanations of each component.

2. Related Work

Educational computer architecture tools traditionally focus on visualization and simulation. MARS is an education-oriented MIPS assembly simulator for undergraduate learning (Vollmar and Sanderson 2006); Ripes provides a visual RISC-V simulator and assembly editor supporting datapath inspection (Petersen 2021); and DrMIPS offers a graphical CPU simulator with single-cycle and pipeline implementations (Nova et al. 2013). These systems are valuable for technical exploration, but their interaction model is simulator-oriented: learners write or inspect code, step through execution, and observe architectural state.

Serious games differ by embedding learning objectives into rules, progression, and feedback cycles (Dörner et al. 2016; Connolly et al. 2012), and recent work reports positive perceptions of serious games in undergraduate computing education (Luccas et al. 2025). Commercial incremental and tycoon games further show the motivational potential of long-term progression, upgrades, and compounding rewards (Hwang and Melcer 2024), a design intent that can be read through procedural rhetoric (Bogost 2007) and Frasca’s view that simulations argue through their rules and outcomes rather than fixed narratives (Frasca 2003). However, such mechanics are rarely mapped directly to CAO concepts such as pipeline, multicycle execution, or multicore processing. Our proposal is positioned between these approaches: rather than replacing precise simulators such as MARS, Ripes, or DrMIPS, it offers a complementary playful environment in which CAO concepts act as functional rules affecting production, speed, cooldowns, and simultaneous processing inside a persistent economy. This raises a trade-off that our full study intends to address: what is gained in engagement and what, if anything, is lost in fidelity relative to direct simulation.

3. Methodology

This work follows a Design Science Research (DSR) approach, suited to building and evaluating an artifact for a recognized problem in context (Hevner et al. 2004). Following the three-cycle view of DSR (Hevner 2007), the relevance cycle addresses the CAO learning difficulty described above, the design cycle comprises the iterative construction and evaluation of *BitCore Tycoon*, and the rigor cycle draws on the serious-game design and evaluation literature. The process followed four stages: (1) problem and requirements definition, including target audience and learning goals; (2) artifact design, formalized in a Game Design Document that records mechanics, progression, and the concept-to-mechanic mapping of Table 1; (3) implementation in Unity/C#; and (4) evaluation, reported here as a preliminary study of the first environment and detailed as a planned controlled study. Generative AI tools were used only to assist with language editing and

LaTeX formatting; all technical content, design decisions, and analyses are the authors' responsibility.

4. Game Proposal

The game is a tycoon/idle serious game for CAO education, in which the player builds computing technology from early components to advanced processor mechanisms across two connected environments: a historical progression loop and a mining/combat loop.

4.1. Historical Progression Loop

In the first environment (Figure 1, left), the player accumulates a resource called Technology, used to unlock components and advance across computing generations. Early items represent basic elements (vacuum tubes, transistors, logic gates, arithmetic units, memory), while later upgrades introduce processor organization and performance mechanisms. Items have educational and economic roles: active items generate Technology per second or per click, passive items multiply production or reduce constraints, and the final item unlocks the next generation. Global research upgrades provide long-term bonuses, and completing a generation archives previous progress and grants a legacy bonus, reinforcing that each generation builds on earlier ones. Figure 1 (right) shows the glossary entry for the vacuum tube, the first available element, providing historical context and technical curiosities to help interested players understand each component.



Figure 1. History pipeline and glossary used to support the game mechanics and terminology.

4.2. Mining and Architecture Loop

The second environment (Figure 2), unlocked at a more advanced generation, introduces an idle combat loop representing simulated cryptocurrency mining. Players assemble and improve rigs from components such as CPU, ALU, RAM, clock, fan, and bus, creating a practical metaphor: better architecture improves mining performance, producing cryptocurrencies that can be reinvested into global research in the first environment. Architecture mechanisms map to gameplay effects: multicycle execution splits work into stages for a stable efficiency bonus; pipeline execution triggers an extra parallel action after a fixed number of automatic attacks; Tomasulo-inspired scheduling reduces cooldowns when no dependency conflicts are present; and multicore processing increases speed or damage and may enable simultaneous attacks against multiple targets. Table 1 summarizes the mapping. Because these are simplified analogies, a tracked design risk is that they may induce misconceptions (for example, of Tomasulo-style scheduling); the

mapping is therefore validated against textbook definitions, and the glossary preserves the precise formulation alongside the gameplay abstraction.

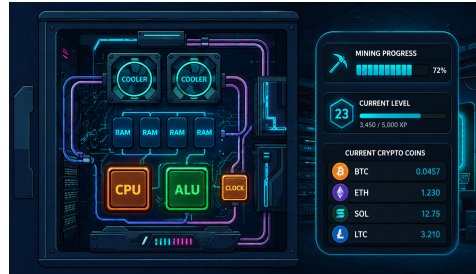


Figure 2. Visual concept of the mining mechanic and cryptocurrency rewards.

Table 1. Mapping between CAO concepts and gameplay mechanics.

Concept	Gameplay representation
Digital components	Unlockable production items that introduce valves, transistors, gates, ALUs, and memory.
Computer Generations	Progression stages with legacy bonuses and increasingly complex components.
Pipeline	Repeated actions trigger parallel routines and extra attacks after a fixed sequence.
Multicycle execution	Work is split into stages, improving efficiency with predictable bonuses.
Tomasulo-inspired scheduling	Dependency-free routines reduce cooldowns and improve action throughput.
Multicore processing	Additional cores increase speed/damage and enable limited simultaneous targeting.

5. Prototype and Technical Implementation

The prototype is developed in Unity using C# and a data-driven architecture, targeting desktop platforms (Windows and Linux) and an audience of computing students and technology enthusiasts. Game entities are defined through configuration assets (item identifiers, costs, levels, dependencies, and production effects), so educational content can be adjusted without rewriting gameplay code and future balancing experiments are supported. The implementation separates the economy, progression, combat, and persistence modules. Critical computed values such as production rate and damage are not stored directly in the save file; instead, the save keeps item identifiers, levels, current generation, active rig, mining progress, and currency values, and derived attributes are recalculated from static configuration on load, reducing inconsistency and supporting future validation of offline progression. The current prototype includes two main screens—the historical laboratory and the mining screen (the latter ongoing)—displaying currencies, generation progress, items and their states, rigs, components, battle progression, cooldowns, and architecture-related effects, with contextual tutorial messages shown on first access to each major feature.

6. Preliminary Evaluation and Study Plan

A preliminary evaluation of the first environment was conducted with seven participants from computing courses, under informed consent and following institutional ethical guidelines for research with human participants, using a seven-item Likert questionnaire (1–5) on fun, motivation, progression interest, historical learning, component didactics, the sequencing puzzle, and the play/learning balance. The composite score was 3.98, above the neutral point (3.0), with no item below neutral. The highest means were for motivation to keep unlocking components and for the didactic value of names, images, and descriptions (both 4.29), followed by component progression (4.14) and fun (4.00). Items concerning historical learning, the sequencing puzzle, and the play/learning balance showed greater response dispersion and less consistently favorable ratings. Preliminary results suggest favorable perceptions of engagement, but they are limited by the small sample and by the evaluation of only the first environment.

Building on this, the full study is planned as a controlled evaluation, following recommendations for assessing digital game-based learning (All et al. 2016). It will combine three kinds of evidence: (i) pre/post knowledge tests with per-concept measures (e.g., data dependencies, throughput, and architecture comparison) stated as measurable hypotheses with a target sample size and success criteria; (ii) validated perception questionnaires such as MEEGA+ (Petri et al. 2019; von Wangenheim et al. 2025); and (iii) gameplay logs (time spent, unlocked components, repeated attempts, and progression bottlenecks) following serious-game analytics approaches (Westera et al. 2014). These logs will also be used to examine whether passive idle progression reduces active engagement with architectural decisions, considering meaningful player actions, upgrade choices, and periods of unattended progression. A central comparison will contrast the game with traditional simulation-based instruction, addressing what an abstract, rule-based representation of CAO gains or loses relative to direct simulation.

7. Final Remarks

This short paper presented *BitCore Tycoon*, an ongoing serious game for learning CAO through tycoon and idle mechanics that maps architectural concepts to progression rules, production bonuses, cooldowns, and simultaneous actions, framed as a Design Science Research artifact whose procedural rhetoric expresses architectural arguments through play. By combining a historical progression loop with a mining/combat loop, and supported by a glossary that preserves precise definitions where mechanics are simplified, the game lets students experience architectural mechanisms as rules that affect system behavior. A preliminary evaluation of the first environment suggests favorable initial perceptions of engagement and didactic potential, though with a small sample. Future work includes completing the second environment, refining the mapping where simplification risks misconceptions, and conducting a controlled study with undergraduate students that compares the game against traditional simulation-based instruction and reports learning gains per concept.

References

- [All et al. 2016] All, A., Castellar, E. P. N., and Van Looy, J. (2016). Assessing the effectiveness of digital game-based learning: Best practices. *Computers & Education*, 92:90–103.

- [Bogost 2007] Bogost, I. (2007). *Persuasive Games: The Expressive Power of Videogames*. The MIT Press, Cambridge, Massachusetts.
- [Connolly et al. 2012] Connolly, T. M., Boyle, E. A., MacArthur, E., Hainey, T., and Boyle, J. M. (2012). A systematic literature review of empirical evidence on computer games and serious games. *Computers & Education*, 59(2):661–686.
- [de Camargo et al. 2025] de Camargo, V. H. S., Junior, M. M. C., da Silva, F. F., and Aylon, L. B. R. (2025). Learning operating systems with educational games: A systematic mapping of the literature. *Journal on Interactive Systems*, 16(1):718–732.
- [Dörner et al. 2016] Dörner, R., Göbel, S., Effelsberg, W., and Wiemeyer, J., editors (2016). *Serious Games: Foundations, Concepts and Practice*. Springer, Cham.
- [Frasca 2003] Frasca, G. (2003). Simulation versus narrative: Introduction to ludology. In Wolf, M. J. P. and Perron, B., editors, *The Video Game Theory Reader*, pages 221–235. Routledge, New York.
- [Hevner 2007] Hevner, A. R. (2007). A three cycle view of design science research. *Scandinavian Journal of Information Systems*, 19(2):87–92.
- [Hevner et al. 2004] Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1):75–105.
- [Hwang and Melcer 2024] Hwang, D. and Melcer, E. F. (2024). Exploring engagement in idle game design. In *2024 IEEE Conference on Games (CoG)*, pages 1–8. IEEE.
- [Larraza-Mendiluze et al. 2024] Larraza-Mendiluze, E., Arbelaitz Gallego, O., Arregi Uriarte, O., Martín Aramburu, J. I., and Lukas Mugika, J. F. (2024). Basic concepts of computer architecture through games and game development. *IEEE Revista Iberoamericana de Tecnologías del Aprendizaje*, 19:195–204.
- [Luccas et al. 2025] Luccas, M. d. S., Pereira, L. T., and Branco, K. R. L. J. C. (2025). Um estudo comparativo de uso de serious games para auxiliar o aprendizado de sistemas operacionais. In *Anais do V Simpósio Brasileiro de Educação em Computação*, pages 165–177. SBC.
- [Ma and Oikonomou 2017] Ma, M. and Oikonomou, A., editors (2017). *Serious Games and Edutainment Applications*. Springer, Cham, 2 edition.
- [Nova et al. 2013] Nova, B., Ferreira, J. C., and Araújo, A. (2013). Tool to support computer architecture teaching and learning. In *Proceedings of the 1st International Conference of the Portuguese Society for Engineering Education*.
- [Null 2023] Null, L. (2023). *Essentials of Computer Organization and Architecture*. Jones & Bartlett Learning.
- [Petersen 2021] Petersen, M. B. (2021). Ripes: A visual computer architecture simulator. In *2021 ACM/IEEE Workshop on Computer Architecture Education (WCAE)*.
- [Petri et al. 2019] Petri, G., Von Wangenheim, C. G., and Borgatto, A. F. (2019). MEEGA+: Um modelo para a avaliação de jogos educacionais para o ensino de computação. *Revista Brasileira de Informática na Educação*, 27(3):52–81.
- [Vollmar and Sanderson 2006] Vollmar, K. and Sanderson, P. (2006). MARS: An education-oriented MIPS assembly language simulator. In *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*, pages 239–243. ACM.
- [von Wangenheim et al. 2025] von Wangenheim, C. G., Petri, G., and Borgatto, A. F. (2025). Using meega/meega+ for the evaluation of educational games: A retrospective. *Revista Brasileira de Informática na Educação*, 33:943–971.
- [Westera et al. 2014] Westera, W., Nadolski, R., and Hummel, H. (2014). Serious gaming analytics: What students log files tell us about gaming and learning. *International Journal of Serious Games*, 1(2).