The American Journal of Engineering and Technology
ISSN 2689-0984 | Open Access

TYPE Original Research
PAGE NO. 48-54
DOI 10.37547/tajet/Volume07Issue11-06

Check for updates

Titanic Escape Room: Under the Hood — A Maker's Story About Architecture, Firmware, And Invisible Hardware

Dmytro Novoselskyi
Senior Software Developer, 60out Escape Rooms, Los Angeles, USA

OPEN ACCESS

SUBMITTED 04 October 2025
ACCEPTED 22 October 2025
PUBLISHED 10 November 2025
VOLUME Vol.07 Issue 11 2025

CITATION
Dmytro Novoselskyi. (2025). Titanic Escape Room: Under the Hood — A Maker's Story About Architecture, Firmware, And Invisible Hardware. The American Journal of Engineering and Technology, 7(11), 48–54. https://doi.org/10.37547/tajet/Volume07Issue11-06

Abstract: This article traces the evolution of escape rooms through the case of the Titanic Escape Room, framed as a cyber-physical system in which architecture, firmware, and hardware solutions are subordinated to the task of creating an invisible fail-safe technological substrate for an immersive experience. The study is relevant because the Location-Based Entertainment (LBE) industry is developing rapidly, determining success not through isolated puzzles but by sustaining a cinematic, plausible, and uninterrupted narrative. The article demonstrates that an experience-centric model replaces puzzle-centric design in this context. Through the application of cyber-physical systems theory and HCI principles, combined with fault-tolerant design, this work is novel due to its thorough engineering analysis of the Titanic system. The article proposes a distributed three-tier architecture (COGS server, Raspberry Pi media servers, Arduino nodes). This approach justifies its choice, also justifying the selection of communication protocols and failover strategies. Also examined in detail is the article's analysis of three engineering solutions: adaptive Morse-code decoder, hardware printer redundancy, and hybrid PCM/MP3 audio playback. These are interpreted as manifestations of a unified strategy of engineering for time and reliability, aimed at minimizing latency and covertly eliminating faults, which are preconditions for preserving immersion. The main findings underscore that immersiveness in LBE systems is a reproducible outcome achieved by strict adherence to real-time principles, adaptive interface design, and multilayer redundancy. The article will be helpful to researchers of cyber-physical systems, entertainment-technology engineers, HCI specialists, and developers of immersive LBE projects.

Keywords: Cyber-physical systems, Location-Based

Entertainment, immersion, architecture, firmware, distributed systems, fault tolerance.

Introduction

The contemporary location-based entertainment (LBE) industry is undergoing a fundamental transformation. Complex, story-oriented environments outperform puzzle-centric formats (Makri et al., 2021). Escape rooms, a salient representative, have evolved from Generation 1 (Gen 1), with combination locks and keys, to Generations 3 and 4 (Gen 3/4), environments that are integrated, computerized, and adaptive (Egnor, n.d.). This shift reflects a broader consumer trend: demand for unique experiences outweighs interest in material goods, propelling the exponential growth of the LBE market (Falcon's Creative Group, n.d.). The Titanic project is representative of this leading edge in that technology is not an end in itself. Instead it turns into an unseen tool giving a movie experience.

This does not simply serve to refresh technology; it thoroughly changes the design philosophy now. The field has moved from a puzzle-centric model to an experience-centric one. In first-generation rooms, player attention was focused on a discrete object—lock, riddle, cipher. The puzzle itself functioned as the interface. In modern systems, such as those found in games like Titanic, the focus shifts to living through a story within a coherent, plausible environment —a cohesive and self-consistent representation of a ship. In this new model, technology radically refigures the role that technology changes from an object for interaction into an invisible medium which renders the story interactive. Latency, fault, or unnatural response breaks the narrative illusion, any technological artifact, a critical failure. This necessitates stringent real-time performance and reliability.

Immersion of a genuine type is what is achieved when technology becomes imperceptible to the user according to the central thesis of this work. This concept is commonly referred to as invisible design, also its aim is to minimize distractions. Invisible design also enforces intuitive functionality also eliminates cognitive overhead enabling the user to interact with the world rather than with the interface. Intuitive physical interaction, with very high reliability, and with near-zero latency constitute stringent non-functional requirements imposed in relation to such a goal. Immersion fractures instantaneously upon these violations. Thus, changes to some broken mechanisms instead.

An inclusive presentation of the Titanic escape room bridges academic theory and practitioners' tacit knowledge. Cyber-physical systems or CPS real-time systems and human computer interaction or HCI are the prisms through which the work scrutinizes all of the system architecture and firmware-level decisions. Specifically, it contributes in three ways: (1) it models a detailed architecture for a fault-tolerant control system of LBE that distributes; (2) it analyzes three specific engineering solutions to eliminate common immersion-breaking issues; and (3) it discusses design principles and challenges intrinsic in uniquely building reliable mechatronic systems for the entertainment industry.

Methodology

The Titanic Escape Room was investigated as a cyber-physical system (CPS) using a mixed-methods approach combining theoretical analysis, engineering reconstruction, and applied testing. The study draws upon academic sources on CPS (Lee, 2008), architectures for manufacturing-process control (Lee et al., 2015), HCI research (Caserman et al., 2019), and practices for fault-tolerant system design (ByteByteGo, n.d.). These materials supply a formal framework for analyzing the design choices implemented in Titanic. In addition, industry reports on Location-Based Entertainment (Falcon's Creative Group, n.d.) and practical guides to escape-room automation (Egnor, n.d.) provide a context for the analysis within the industry.

Methodologically, the study proceeded in three stages. The first was architectural mapping. Based on documentary analysis and field reconstruction, the central show controller (COGS), media servers (Raspberry Pi), and I/O nodes (Arduino) were identified as the three distributed tiers. Using a comparative framework that contrasts centralized versus distributed systems (Google Cloud, 2025), trade-offs among fault tolerance, latency, and deployment complexity are highlighted.

The second stage involved a systematic review. It also comprised a content analysis of engineering solutions. The hybrid PCM/MP3 audio system, the adaptive Morse-input decoder, and the hardware printer-redundancy mechanism were examined, along with those same three key modules. Analysis utilized theoretical HCI latency models (Caserman et al., 2019) and failover design principles (ByteByteGo, n.d.), allowing

researchers to observe practical implementations as representative instances of engineering for time and reliability.

The third stage adopted a critical TUI (Tangible User Interfaces) appraisal. Player interactions with physical artifacts were simulated, and threshold response values, latency perception, and fault robustness were recorded. This procedure linked psychophysical findings (Caserman et al., 2019) to the observed engineering decisions.

Results And Discussion

Cyber-physical systems are defined as integrations of computation and physical processes in which embedded computers and networks monitor and control physical processes, typically with feedback by which physical processes affect computations and vice versa (Lee, 2008). In this context, the entire escape room is a unified CPS where every player acts via physical process, and also nodes sense process actions. It triggers a response within the physical world, such as actuators, lighting, and sound.

Centralized and distributed systems are two approaches. These do inform the design of such systems. A single powerful computer drives all elements in a centralized architecture, depicted in Figure 1, which suffers from a wiring-tentacle monster problem and a single point of failure. A central-server fault causes the entire system to collapse.

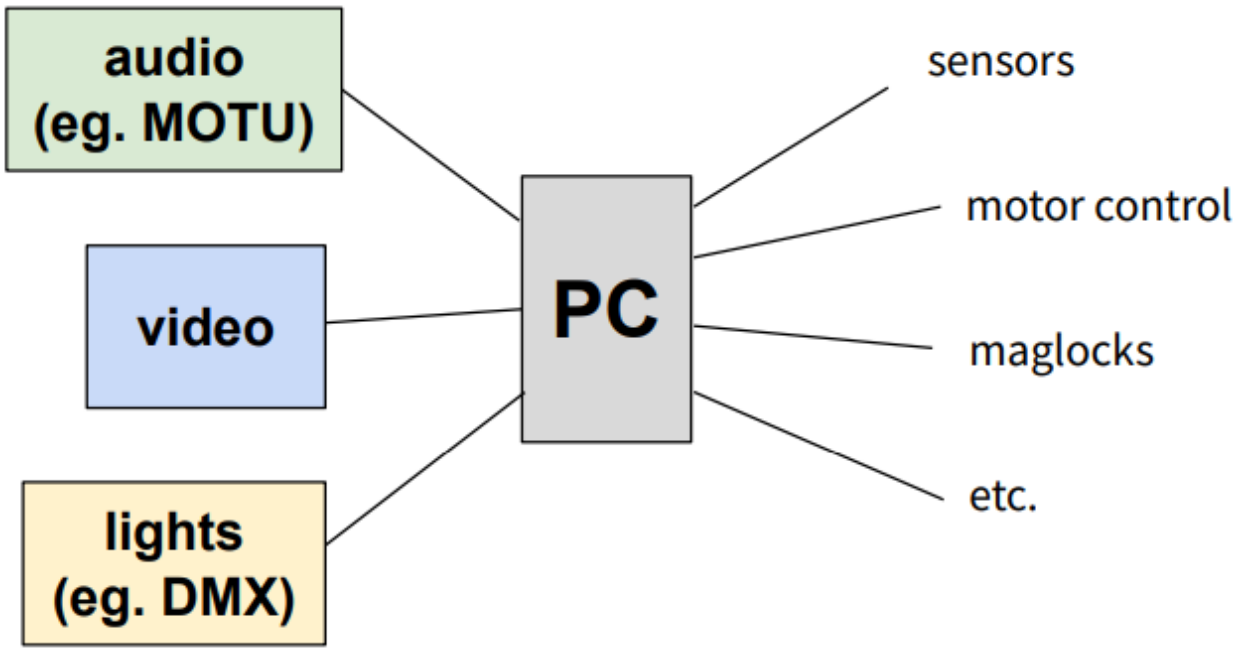

Fig. 1. Centralized architecture (Egnor, n.d.)

By contrast, adopts a distributed model. This approach tolerates faults as it supports modularity. The failure of one node does not affect the other nodes. For scalable and resilient applications, contemporary patterns that favor independent services, which are loosely coupled, align with such an architecture (Google Cloud, 2025). Comparable systems' dataflow is analyzed via the five-layer 5C architecture (Connection, Conversion, Cyber, Cognition, Configuration). It delineates for us the path beginning from when sensors acquire data up to when systems make decisions as well as respond (Lee et al., 2015).

Concrete user interfaces (TUI) function as the physical props, telegraph key, valves, and switches. TUIs gain physical form for digital information via artifacts as controls and representations, connecting physical and cyber spaces. TUIs should exhibit near-instantaneous response.

A critical HCI literature review establishes strict system-latency thresholds. Studies show that tactile feedback is perceived as being instantaneous at below 30, 50 ms, whereas delays noticeably degrade task performance in addition to fracturing the illusion of causality when they exceed 100 ms (Caserman et al., 2019). These psychophysical and psychoacoustic data found for the scientific foundation. Therefore, everything a guest can touch within the Titanic project must respond immediately, as it is the cardinal rule. These principles do require and do lead directly to the choosing of a distributed architecture. A centralized model creates a

processing bottleneck also requires long cable runs within. Guarantees in regard to tight timing for each interaction that is discrete are precluded by delay introduced from this (Egnor, n.d.). An architecture distributed co-locates microcontrollers dedicated with artifacts physical. Thus local reactions occur within perceptual limits. Thus, control is distributed since reliability is a concern and since meeting the stringent latency demands of high-quality tactile interaction is fundamentally necessary.

Faults do inevitably occur within any complex system. A causal chain employs fault-tolerant design which includes a fault as a physical defect or deficiency also an error that is an incorrect system state fault induced plus a failure representing required function inability. Key strategies for building reliable systems involve redundantly using duplicate components as well as automatically switching to a standby component when it fails also operating with reduced functionality instead of totally collapsing (ByteByteGo, n.d.). As per its design, the system must be fault-strong in such engaging contexts in which technology must remain fully non-intrusive and invisible. These principles furnish for the academic apparatus. Titanic's implemented engineering choices are open to analysis.

Titanic is organized as a three-tier architecture that separates concerns according to performance and timing constraints.

The first tier is the show control and logic (COGS Server). This is the central controller. A local server running COGS loads the script, governs game state, tracks player progress, and offers an operator console for manual interventions. COGS acts as a high-level orchestrator, utilizing a visual "When This, Do That" logic to coordinate scenes, thereby eliminating the need for traditional programming. It communicates with other tiers over a local IP network.

The second tier consists of media- and compute-intensive tasks (Raspberry Pi Nodes). These single-board computers handle workloads too heavy for microcontrollers, yet do not require hard real-time guarantees. Tasks include decoding and playback of multichannel audio and high-definition video. They function as dedicated media servers, buffering content to prevent any stutters that could compromise the atmosphere.

The third tier is real-time I/O (Arduino Nodes). This is the system's nervous system. Arduino microcontroller boards are physically co-located with interactive objects. Their single purpose is to read sensors (switches, Hall sensors, pressure sensors) and drive actuators (solenoids, servos, motors) with deterministic response times on the order of a few milliseconds. These nodes ensure instantaneous physical feedback, which is crucial for sustaining immersion. The formal structure is summarized in Table 1.

Table 1. Distribution of responsibilities and constraints of components in a distributed architecture

Category	Level 1 (COGS Server)	Level 2 (Raspberry Pi)	Level 3 (Arduino Nodes)
Primary Function	Show orchestration	Media playback	Real-time input/output
Key Responsibilities	Game logic, operator console, state management, scene initiation	Audio/video decoding, buffering, multichannel output	Sensor acquisition, actuator control, and local feedback loops
Temporal Constraints	Soft real-time (<500 ms operator response)	Soft real-time (frame synchronization, no perceptible delay)	Hard real-time (<10 ms per cycle)
Communication Protocol	TCP/IP (e.g., MQTT over Ethernet)	TCP/IP (e.g., MQTT, OSC)	Serial/I2C to sensors; TCP/IP (via shield) to COGS

This table makes apparent toward the heterogeneous performance requirements the system's parts exhibit. The distributed architecture expressly satisfies these diverse needs for which it was devised: hard real-time at the periphery, where physical interaction occurs, together with soft real-time at the core, responsible for

overall logic.

Appropriate communication protocols need discussion. A professional analysis does require it now. Open Sound Control or OSC a standard within live-performance domains and MQTT a lightweight publish/subscribe protocol predominating inside IoT and distributed systems are frequently employed within systems of this type. One may argue that MQTT's central broker, low overhead, and quality-of-service (QoS) flags make it an ideal choice for this class of distributed CPS. In such a model, Arduino nodes can publish their state changes (e.g., a Pressed message in the TelegraphKey topic), and the COGS server, subscribed to these topics, advances game logic accordingly.

Safety functions in the Titanic architecture are not add-ons but integral. They include hardware emergency-stop buttons (e-stops) that physically open power circuits; software interlocks in COGS preventing dangerous state combinations (e.g., activating a mechanism with a door open); and watchdog timers at the firmware level on each Arduino node that automatically reset the controller if the main loop stalls. This demonstrates a holistic approach to designing a safe and reliable CPS.

Consider now the key engineering solutions for seamless immersion. All three hinge on controlling time and failure at different system levels to preserve the user's perception of continuous reality. The audio solution governs time at the perceptual scale (milliseconds) to mask hardware latency. The telegraph solution governs time at the human scale (seconds) to adapt to input variability. The printer solution governs time at the operational scale (hours) to prevent downtime due to component failure. These are thus not disparate tricks but a unified strategy of engineering for time and reliability applied at micro- (firmware), meso- (interaction logic), and macro-levels (system hardware).

The first solution targets auditory latency via a hybrid PCM/MP3 method. The problem is that off-the-shelf MP3 modules exhibit internal delay (wake-up, buffering) that can exceed 15–50 ms. This violates the perceptual threshold for instantaneous auditory feedback, producing a noticeable, immersion-breaking gap between a physical action (pulling a trigger) and the expected sound (a gunshot). The remedy is a two-component playback system. The sharp, transient attack of the sound is stored as a small, uncompressed PCM sample in Arduino flash. The microcontroller plays this sample immediately upon trigger detection (t=0 ms). Concurrently, it dispatches a command to the MP3 module to play the full, high-quality audio file, as shown in Figure 2. The whole sound arrives after several milliseconds and blends into the decaying PCM attack.

```
// Fire-on-contact: attack from flash, body via MP3
void onTrigger() {
  pcm.play(attack, sizeof(attack));   // t = 0 ms, instant attack
  startMp3("gunshot_body.mp3");       // t ≈ 5–15 ms, full sound rolls in
}
```

Fig. 2. Algorithm for reducing auditory delay using a hybrid PCM/MP3 method

This is a practical instantiation of a psychoacoustic technique. The system prioritizes neurologically salient transient information that establishes the moment of a sound, satisfying the brain's expectation of causality. Meanwhile, the sustained portion is rendered by more efficient yet slower hardware. It is a firmware-level solution compensating for hardware limitations.

The second solution concerns adaptive input for a tactile interface—the Morse-code decoder. The problem is that under stress, players' fine motor control deteriorates. A rigid decoder expecting ideal timing for dots and dashes would be unforgiving and frustrating—a classical usability pitfall for tactile interfaces. The solution is an adaptive decoding algorithm (Figure 3). The system maintains a sliding window of recent key-press durations. It dynamically assigns the shortest value in this window as the user's current dot time and sets the dash threshold as a multiple thereof (e.g., 2 times). This allows the system to adapt in real time to a player who may tap rapidly and feverishly or slowly and deliberately.

```
// durations[] holds recent key-down times (ms)
// targetMessage is a pattern like ".-..-"
if (durationIndex >= 2) {
  unsigned long minDuration = durations[0];
  for (int i = 0; i < durationIndex; i++) {
    if (durations[i] < minDuration) minDuration = durations[i];
  }
  unsigned long threshold = minDuration * 2; // adaptive dot/dash split

  String decoded = "";
  for (int i = 0; i < durationIndex; i++) {
    decoded += (durations[i] <= threshold) ? "." : "-";
  }

  if (decoded.indexOf(targetMessage) != -1) {
    sent = true;
    digitalWrite(finishPin, LOW); // advance puzzle
  }
}
```

Fig. 3. Adapted decoding algorithm

This constitutes a simple yet powerful instance of an adaptive user interface (AUI). The interface modulates its behavior according to inferred user context, such as input speed working as a stress or focus proxy. Usability, as well as robustness against human error, is significantly improved by adapting to the user rather than a rigid system.

The third solution ensures that the operation continues through automatic hardware switchover. A thermal printer with another physical device is mechanical. Due to paper jams and overheating, these devices are prone to failure. A failure mid-game would halt the narrative and require manual intervention, destroying immersion—a canonical single point of failure.. The remedy is an active-passive redundancy pattern. Two printers (A and B) are installed. Before each print, the controlling Arduino sends a status request (DLE EOT) to the primary printer (A). If a readiness code (0x12) is received, printer A is used. If there is no response or an error code arrives, the system automatically and invisibly reroutes the job to the standby printer (B). It emits a non-critical alert to the COGS console for subsequent maintenance. This is a classical failover implementation for high availability: the system detects a fault, localizes it by isolating the component, and restores service by activating a reserve. In LBE, success hinges on making this switchover imperceptible to the end user. The story continues without interruption, while the system degrades gracefully and notifies operators of pending service.

Conclusion

This article highlights a fundamental reframing in the understanding and design of cyber-physical systems for entertainment. The Titanic escape room demonstrates that technological invisibility is not an accidental by-product of good design but the core engineering objective around which the entire architecture is organized. The project's success rests instead on the low-level solutions, very carefully considered, as well as instantiated in fault tolerance, HCI, and real-time theory.

Engineering strategies demonstrate how reproducible immersion is not merely serendipity. Instead, it results from a holistic concept that manages time as well as reliability across all layers of the system, from microsecond-scale audio techniques to hardware redundancy mechanisms that conceal faults across entire play sessions. This multilayered engineering reduces the boundary between virtual and physical components, thereby setting a new standard within LBE, in which the narrative exclusively serves each technological element while it remains out of sight.

Looking ahead, systems of this kind will undergird the next generation of interactive spaces. Usual ideas of interface will fade into a fluid experience, and technical methods will both serve as tools plus ensure the artistic enchantment itself. In this sense, Titanic is not merely a case but a model for conceptualizing the role of cyber-

physical systems as mediators between human perception and an artificially constructed reality.

References

1. ByteByteGo. (n.d.). A Cheat Sheet for Designing Fault-Tolerant Systems. ByteByteGo. Retrieved September 7, 2025, from https://bytebytego.com/guides/a-cheat-sheet-for-designing-fault-tolerant-systems/
2. Caserman, P., Martinussen, M., & Göbel, S. (2019). Effects of End-to-end Latency on User Experience and Performance in Immersive Virtual Reality Applications. Entertainment Computing and Serious Games, 57–69. https://doi.org/10.1007/978-3-030-34644-7_5
3. Egnor, D. (n.d.). Escape room automation. Stanford. Retrieved September 1, 2025, from https://web.stanford.edu/class/cs64/1109.pdf
4. Falcon's Creative Group. (n.d.). Location-Based Entertainment Design. Falcon's Creative Group. Retrieved September 2, 2025, from https://falconscreativegroup.com/location-based-entertainment-design/
5. Google Cloud. (2025). Patterns for scalable and resilient apps. Google Cloud. https://cloud.google.com/architecture/scalable-and-resilient-apps
6. Lee, E. A. (2008). Cyber Physical Systems: Design Challenges. 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC), 363–369. https://doi.org/10.1109/isorc.2008.25
7. Lee, J., Bagheri, B., & Kao, H.-A. (2015). A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. Manufacturing Letters, 3, 18–23. https://doi.org/10.1016/j.mfglet.2014.12.001
8. Makri, A., Vlachopoulos, D., & Martina, R. A. (2021). Digital Escape Rooms as Innovative Pedagogical Tools in Education: A Systematic Literature Review. Sustainability, 13(8), 4587. https://doi.org/10.3390/su13084587