



LAMPORT GENERAL SYSTEMS

A game concept by Jack Drabenstadt

AT ITS CORE

- *Lamport General Systems* is a first-person immersive sim where you play as a lone technician in a derelict facility, manipulating a self-repairing network of robots and computers that has become corrupted while uncovering the consequences of corporate neglect and deciding the fate of a mysterious AI.



PLAYER EXPERIENCE GOALS

- **An Impossible Task**

- Feel the crushing responsibility of being the lone technician tasked with reviving or condemning a vast, decaying system.

- **Strategic Infiltration**

- Choose between stealth, brute force, clever exploits, or low-level coding to overcome hostile machines and networks.

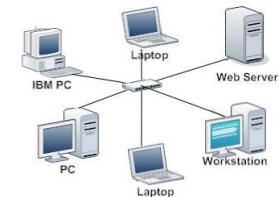
- **Systemic Awe**

- Experience both awe and dread at the network's uncanny resilience and ability to change and adapt separately from the player's actions.

- **Satisfaction through Systemic Learning**

- Develop an increasing understanding of how the network functions, the tools available to correct or disrupt parts of it, and a sense that you can make decisions about how to interact with the system based on this knowledge.

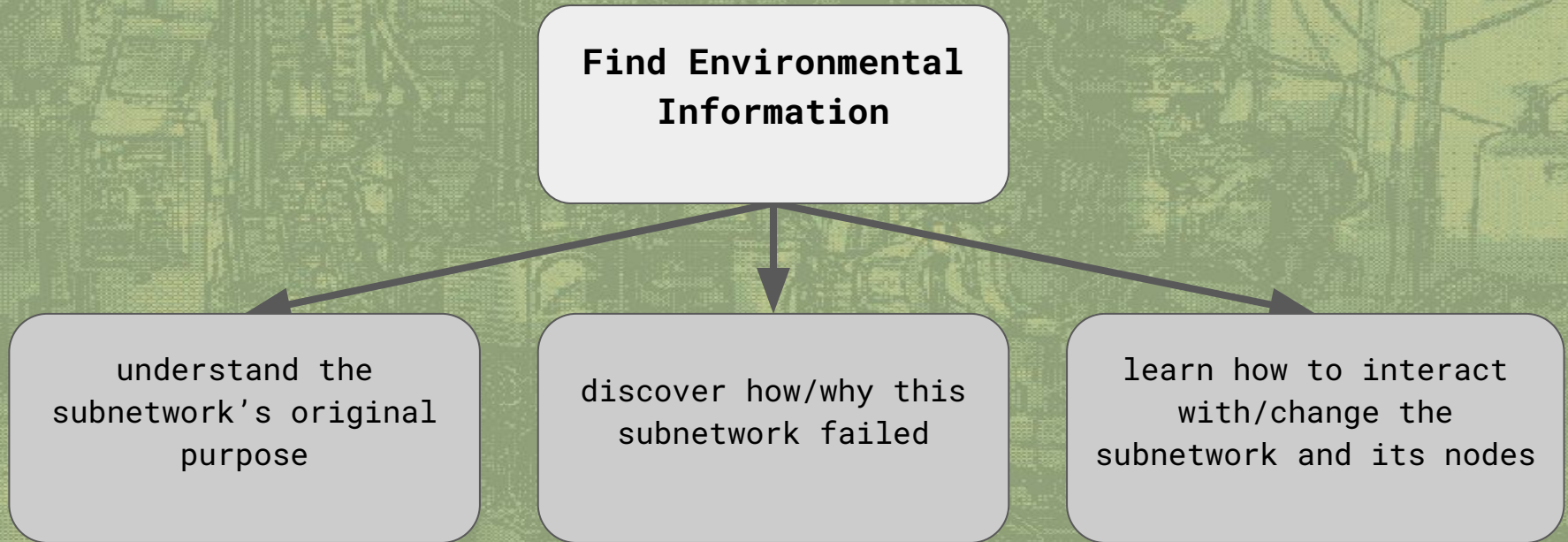
```
pushq    %rbp
movq     %rsp, %rbp
pushq    %r12
pushq    %rbx
subq     $32, %rsp
movl     %edi, -20(%rbp)
movq     %rsi, -32(%rbp)
movq     %rdx, -40(%rbp)
movq     -40(%rbp), %rdx
movq     -32(%rbp), %rcx
movl     -20(%rbp), %eax
movq     %rcx, %rsi
movl     %eax, %edi
call     M2_init
movq     -40(%rbp), %rdx
movq     -32(%rbp), %rcx
```



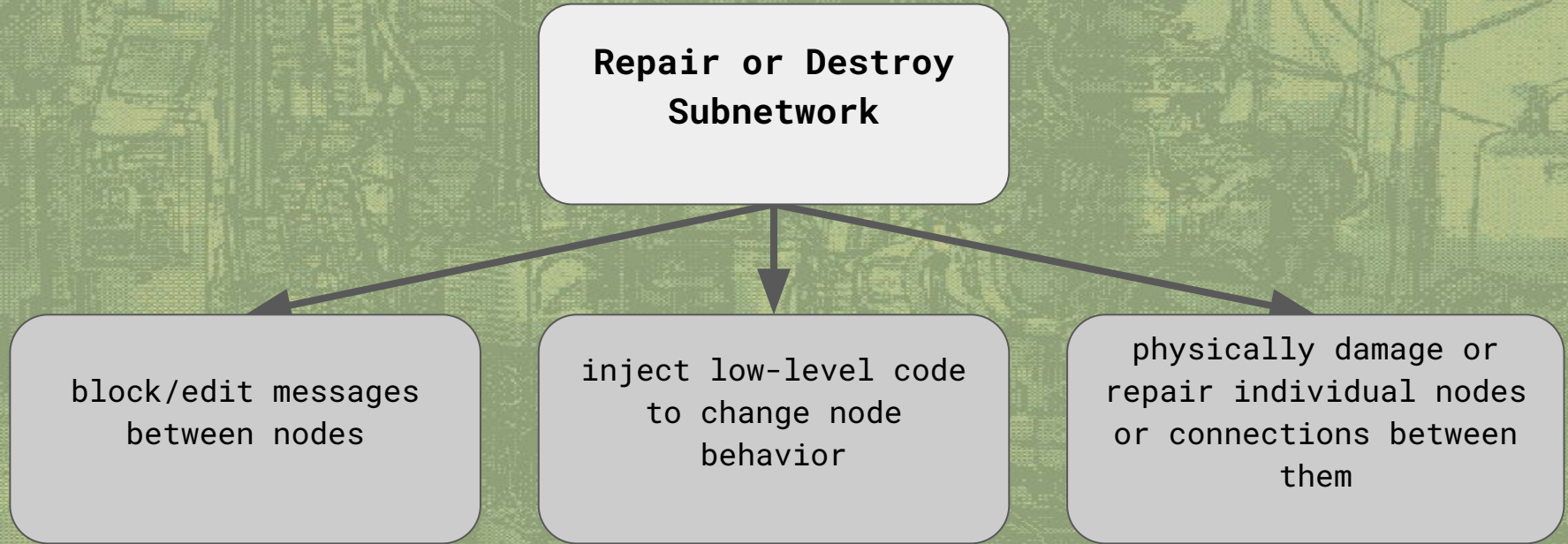
CORE LOOP



CORE LOOP



CORE LOOP



MECHANICS

- **Message Manipulation**
 - Intercept, edit, replay, or manufacture messages and instructions sent across the network to control nodes.
- **Instruction Injection**
 - Temporarily patch or further corrupt nodes with low-level code.
- **Node Sabotage**
 - Cause nodes to fail or isolate them physically (power cuts, broken links).
- **Observation & Stealth**
 - Hide, move carefully, and avoid detection by humanoid robots or the network itself.
- **Emergent Interaction**
 - player actions ripple through the network and produce surprising outcomes.



DESIGN INSPIRATIONS

- **Byzantine Fault Tolerance (BFT):**
 - Distributed systems that continue functioning despite node failures or inconsistent behavior—inspires a resilient network that degrades, repairs, and mutates over time with or without intervention.
- **Immersive Sims and Emergent Systems:**
 - Games like Deus Ex demonstrate systemic design where independent features interact dynamically—inspires non-deterministic gameplay where players discover solutions the developers never anticipated.
- **Interactivity, Emergence, and Narrative Design:**
 - Systems like Façade's behavior system allow agents to interact in non-scripted ways—inspires network nodes that communicate and respond to each other, creating opportunities for players to manipulate or disrupt these interactions.



STORY & THEMES

- You are a lone technician sent by your company Lamport General Systems to restore or, if you choose, condemn the network of nodes. Logs and traces reveal the system's forgotten purpose and why it began failing. Deeper inside, you discover an anomalous AI potentially altering your motivations in your choice to repair or destroy the network.
- **Themes**
 - **Corporate Neglect** – You're sent alone with minimal tools to manage an impossible job reflecting how corporate systems neglect individual workers.
 - **Systemic Resilience** – BFT Networks adapt, resist, and never die easily.
 - **Emergence** – Your manipulations spark cascading, unpredictable outcomes in a dynamically changing system.
 - **Moral Reckoning** – Learning the system's original purpose, the causes of its failure, and the hidden AI alters how you weigh the choice between repairing or sabotaging the network.

DEVELOPMENT APPROACH

- **Engine**

- Godot – lightweight, open-source, with a signal system which is potentially ideal for modeling networks.

- **Research**

- Researching Byzantine Fault Tolerant systems and the underlying mechanics of Facade to inform network design and player interaction.

- **Prototyping**

- Physically prototype the game to compare and balance network behaviors and player interaction.

- **Deliverable**

- Vertical slice showcasing a functioning network with multiple interacting nodes and ways for the player to impact the network.

- **Risks**

- Emergent system behaviors may require extensive tuning.
- Balancing realism vs. clarity: making low-level programming engaging without overwhelming players.

RECAP

- Players manipulate a network through exploration and tactical decisions.
- Actions create meaningful, cascading consequences across the system.
- The network operates autonomously—degrading, adapting, and repairing itself.