# CISC 322/326

ScummVM

A3 Enhancement proposal Presentation.

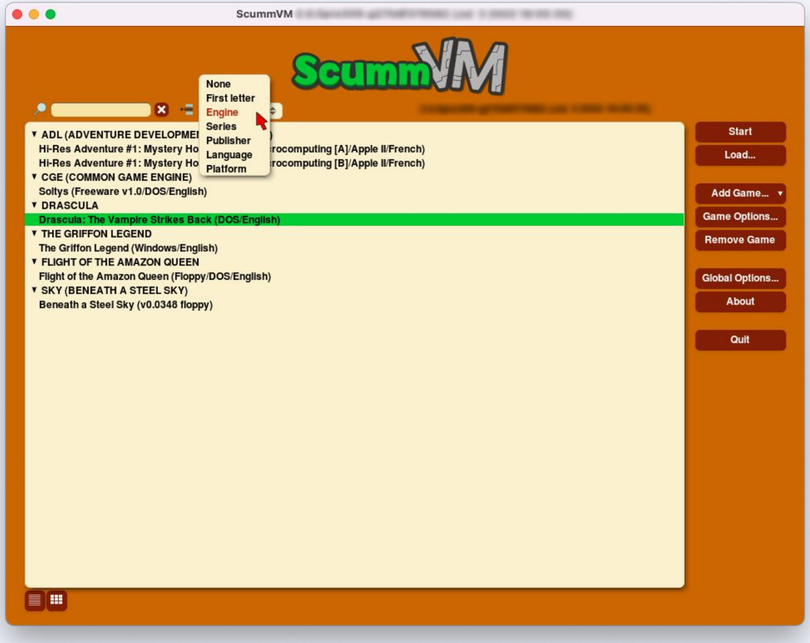
YouTube Link: https://youtu.be/xWtk4thjtME

# Group Roles

# Group 2: MAWLOK

| | |
|---|---|
| **Owen Meima** | Team lead |
| **Andrew Zhang** | Presenter |
| **Michael Marchello** | Presenter |
| **Kevin Panchalingam** | Developer |
| **Lance Lei** | Developer |
| **Zhuweinuo Zheng** | Developer |

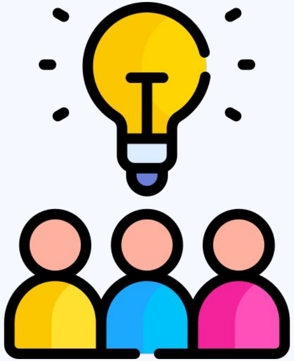
# Introduction


ScummVM
None
First letter
Engine
Series
Publisher
Language
Platform
ADL (ADVENTURE DEVELOPME
Hi-Res Adventure #1: Mystery Ho rocomputing [A]/Apple II/French)
Hi-Res Adventure #1: Mystery Ho rocomputing [B]/Apple II/French)
CGE (COMMON GAME ENGINE)
Soltys (Freeware v1.0/DOS/English)
DRASCULA
Drascula: The Vampire Strikes Back (DOS/English)
THE GRIFFON LEGEND
The Griffon Legend (Windows/English)
FLIGHT OF THE AMAZON QUEEN
Flight of the Amazon Queen (Floppy/DOS/English)
SKY (BENEATH A STEEL SKY)
Beneath a Steel Sky (v0.0348 floppy)
Start
Load...
Add Game...
Game Options...
Remove Game
Global Options...
About
Quit

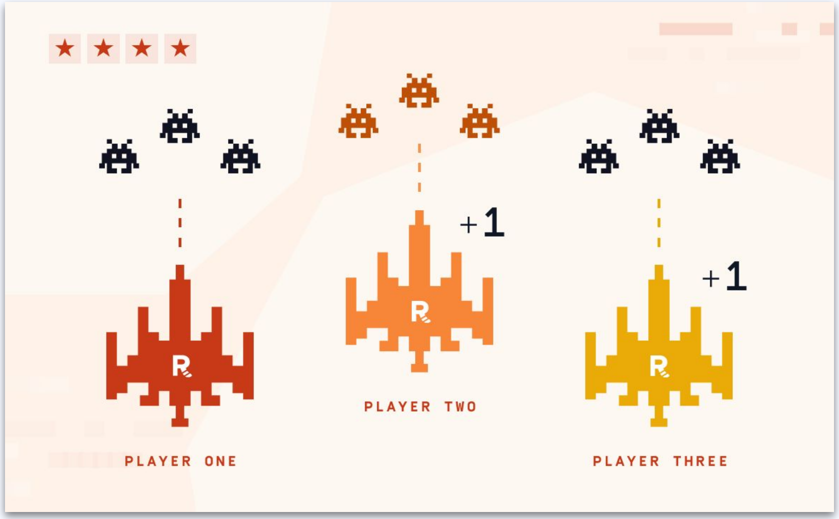
- Explored ScummVM application, concrete architecture, and conceptual design, including its architectural style and components.
- Identified areas for improvement, focusing on enhancing the user experience for game creation.
- Decided to add a multiplayer backend framework to simplify server joining and support developers in implementing multiplayer features.

# Derivation Process

- We initially brainstormed a game creator feature for ScummVM, inspired by Scratch, but abandoned it due to implementation complexity.
- Decided to implement a backend multiplayer framework to simplify future development of multiplayer features.
- Reviewed past reports, studied reference architectures like Minecraft, and created sequence diagrams to map the framework integration with ScummVM components.



# Multiplayer Framework

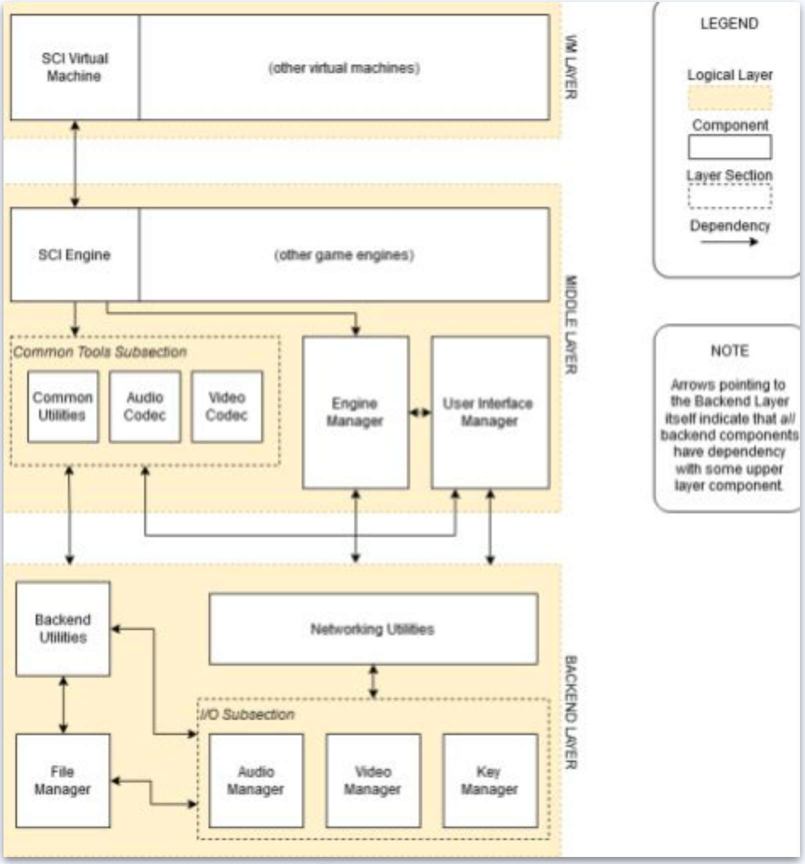


- The multiplayer framework is a backend layer component, enabling developers to integrate multiplayer functionality without building it from scratch.
- It is managed by the Osystem API, a superclass overseeing most backend layer components, though not a component itself
- The previous presentation clarified the rationale for removing the Osystem API.

# Testing the Impact of Interactions

- Planned to test the new multiplayer feature by creating an example multiplayer game and hosting it on a ScummVM server to examine how the different components and layers would interact.
- This testing would uncover how seamlessly the game engine integrates with the new Networking Utilities, whether the common utility files are sufficient for networking requirements, and how the user-facing UI components interact with the underlying networking functionality.
- The testing would also reveal performance implications, scalability concerns, and how the system handles edge cases like network failures, latency issues, and unexpected player behaviors.

# Potentially Impacted Subsystems



# SEI SAAM Architectural Analysis

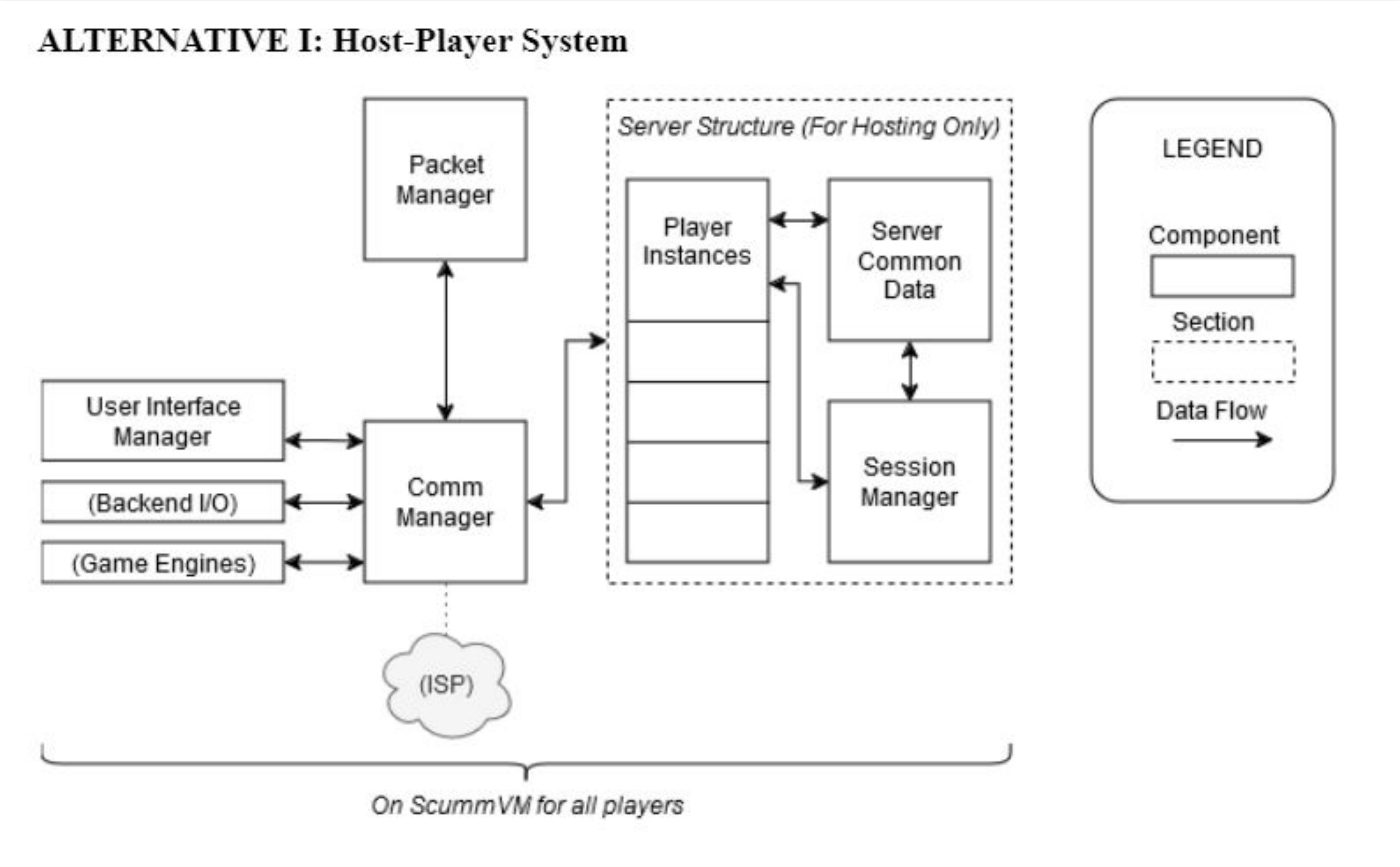
## Major Stakeholders

- Game/Engine Developers
- Players
- Server Hosts

## Non-Functional Requirements

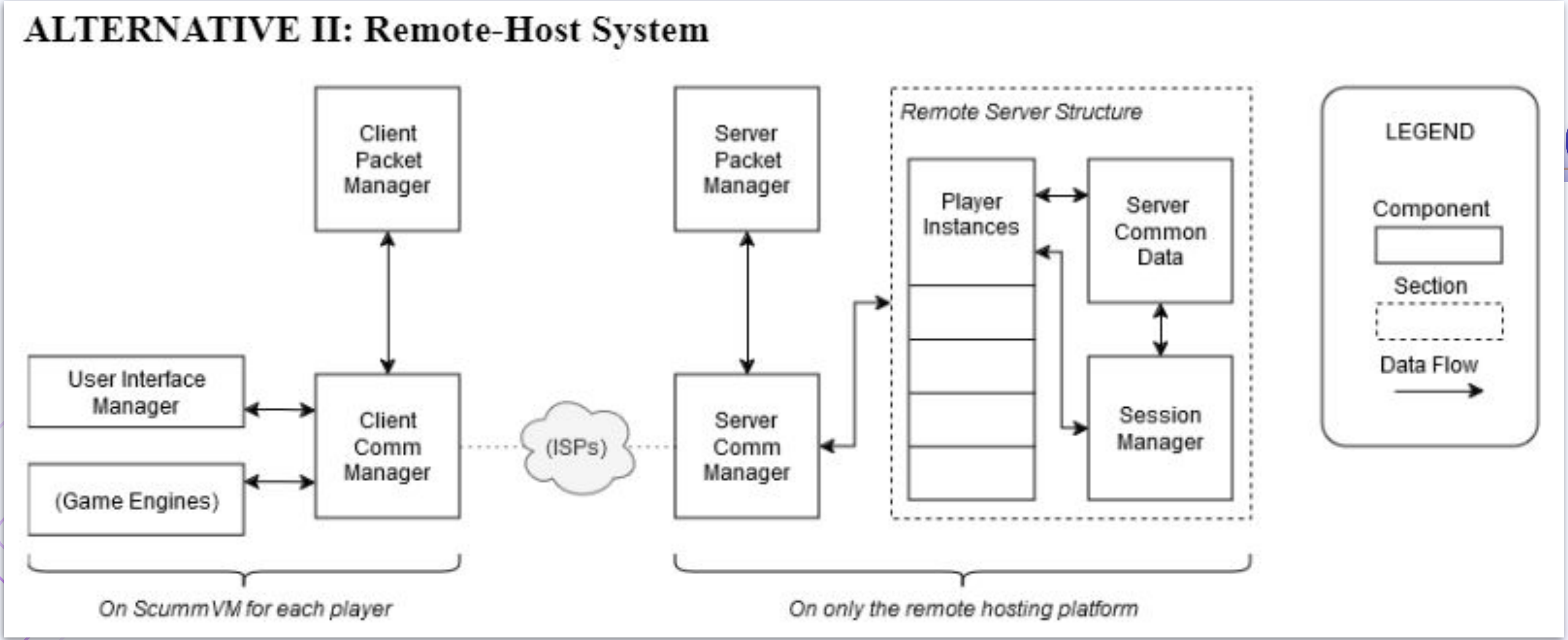
- Game/ Engine Developers:
  - Maintainability
  - Testability
  - Evolvability
- Players:
  - Usability
  - Performance
  - Availability
- Server Hosts:
  - Security
  - Performance
  - Availability

# Alternative Enhancement 1



ALTERNATIVE I: Host-Player System

# Alternative Enhancement 2



ALTERNATIVE II: Remote-Host System

# Best Alternative Enhancement

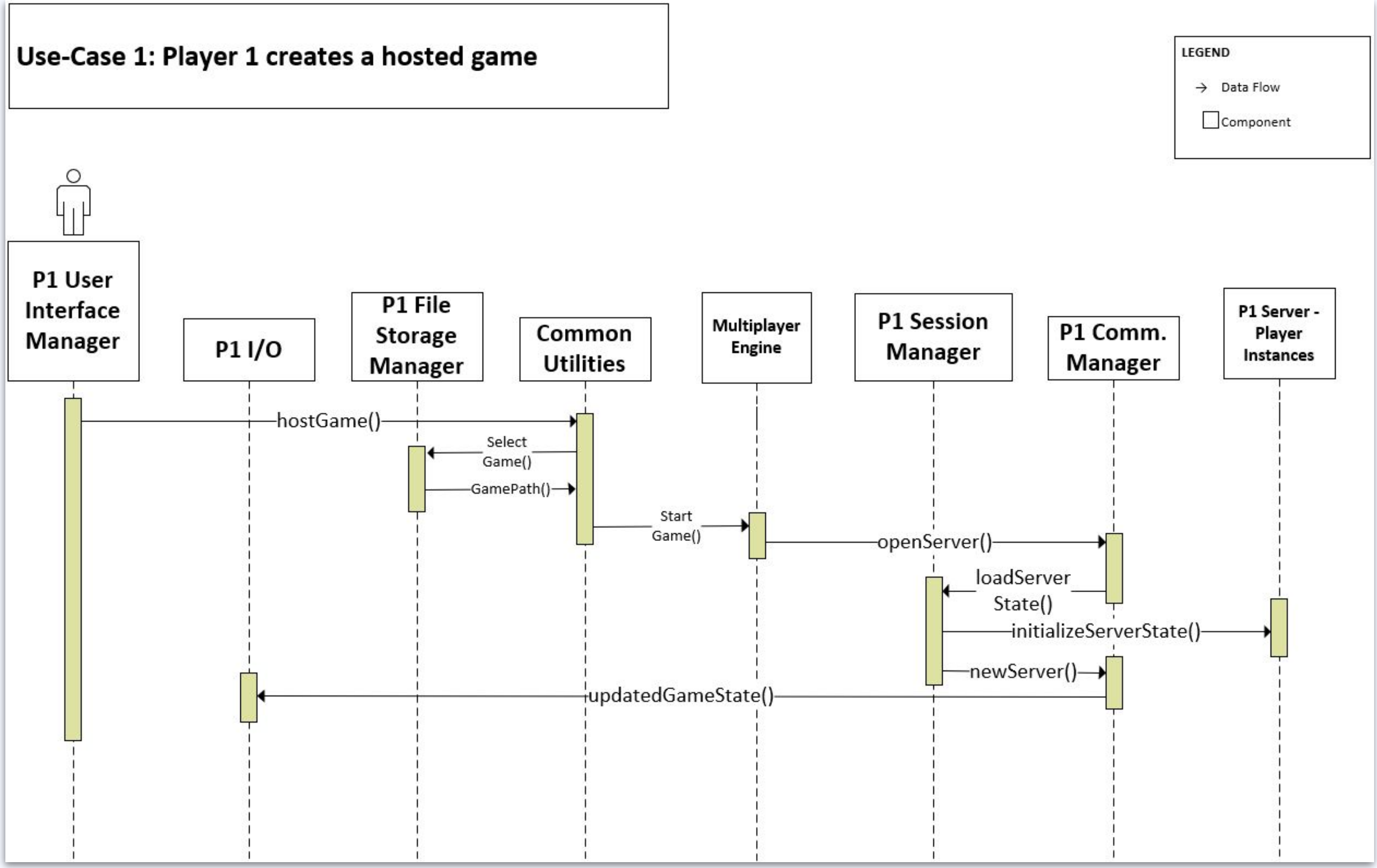
## CHOSEN ALTERNATIVE: HOST-PLAYER SYSTEM

- Disadvantages are not as catastrophic.
- Has more advantages than alternative two.

# Potential Risks

- **High Pressure on the Server**: In multiplayer games, the enhancement system is designed to process inputs and leave the task of handling them to the server. The server then sends the resulting game data back to every client, which can place significant pressure on the server.
- **Potential for Bottlenecking and Reduced Connection Quality**: The server may experience bottlenecks if it is overwhelmed with too much data, potentially leading to a reduction in connection quality and slower game performance.

# Sequence Diagram



Use-Case 1: Player 1 creates a hosted game

# Lessons Learned

- SEI SAAM Architectural Analysis.
- How component relationships change thanks to implementation of new features.
- Network communication.

SAAM

SOFTWARE ARCHITECTURE
ANALYSIS METHOD

# Conclusion

- Analyzed the ScummVM architecture and identified possible improvements, ultimately deciding to add a multiplayer framework after brainstorming viable ideas.
- Potential interactions of the new feature were explored, focusing on its impact from high-level to low-level design perspectives.
- Stakeholders and non-functional requirements were evaluated, with sequence diagrams visualizing usage and risks associated with the framework identified.
- The project demonstrated the ability to create and integrate new features into software, providing insights into ScummVM architecture and functionality.

Merry Christmas

WHYYYYYY WEREN'T YOU AT ELF PRACTICE!?
imgflip.com