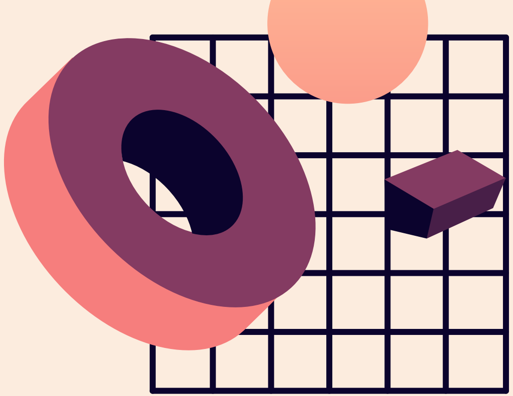


YouTube Video Link

https://youtu.be/klEbpnNFlnA

Assignment #3

Group #25 - 2024/12/02 - CISC 322/326

Group Lead: Elill Mathivannan

Presenters: Henry Xiu, Amaan Javed

Members: Momin Alvi, Elias Frigui, Ahmad Tahir

Stakeholders

1) **Users**
    - Usability, Accessibility, Privacy
2) **Developers**
    - Maintainability, Testability, Scalability
3) **Community contributors**
    - Modularity, Extensibility, Debugging, Good Documentation

New Feature & Motivation

- Accessibility
- Enhanced Player Experience
- Modernization
- Context-Aware Design

Enhancement Architectural Styles

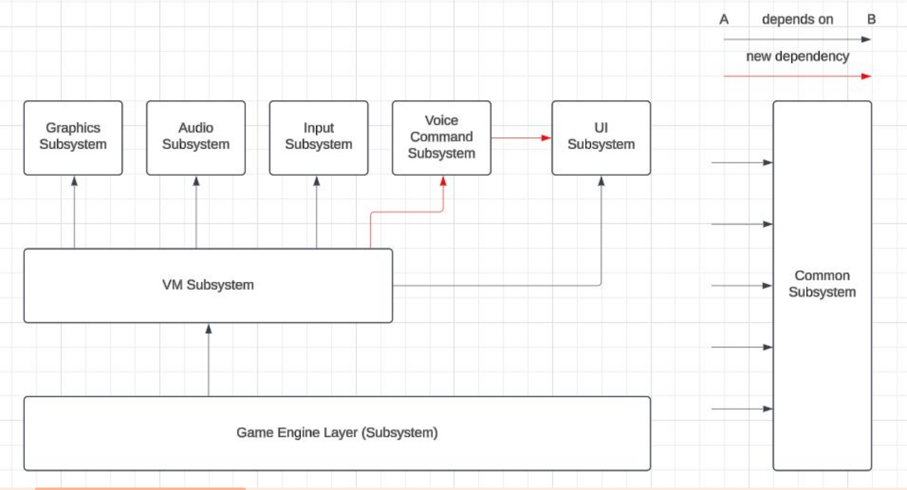
1) **Modularity & Layers**
   - Separation of concerns
2) **Feedback Control System**
   - Ensures commands are valid while giving live feedback

Impacted Subsystems

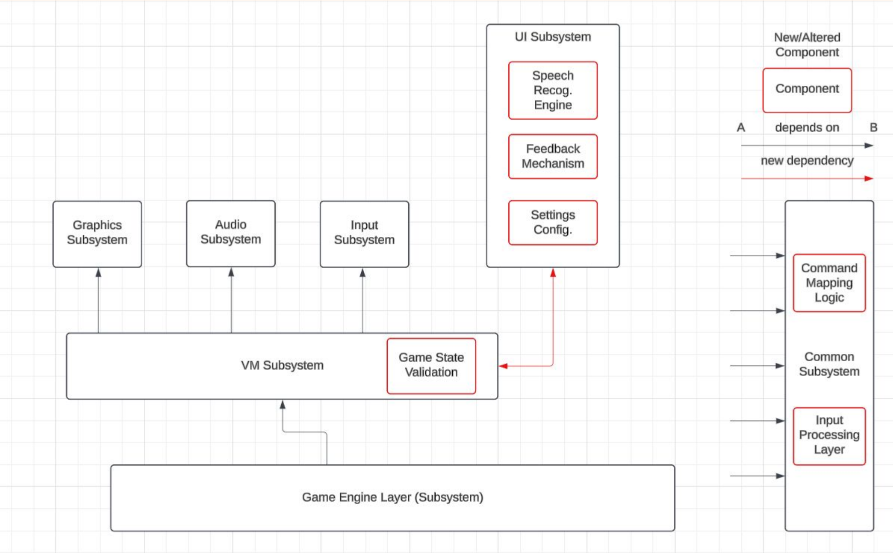
- Speech Recognition Engine
- UI Subsystem
- Event Manager
- Game State Validation

Enhancement Realization

1) Standalone Subsystem



2) Embedded



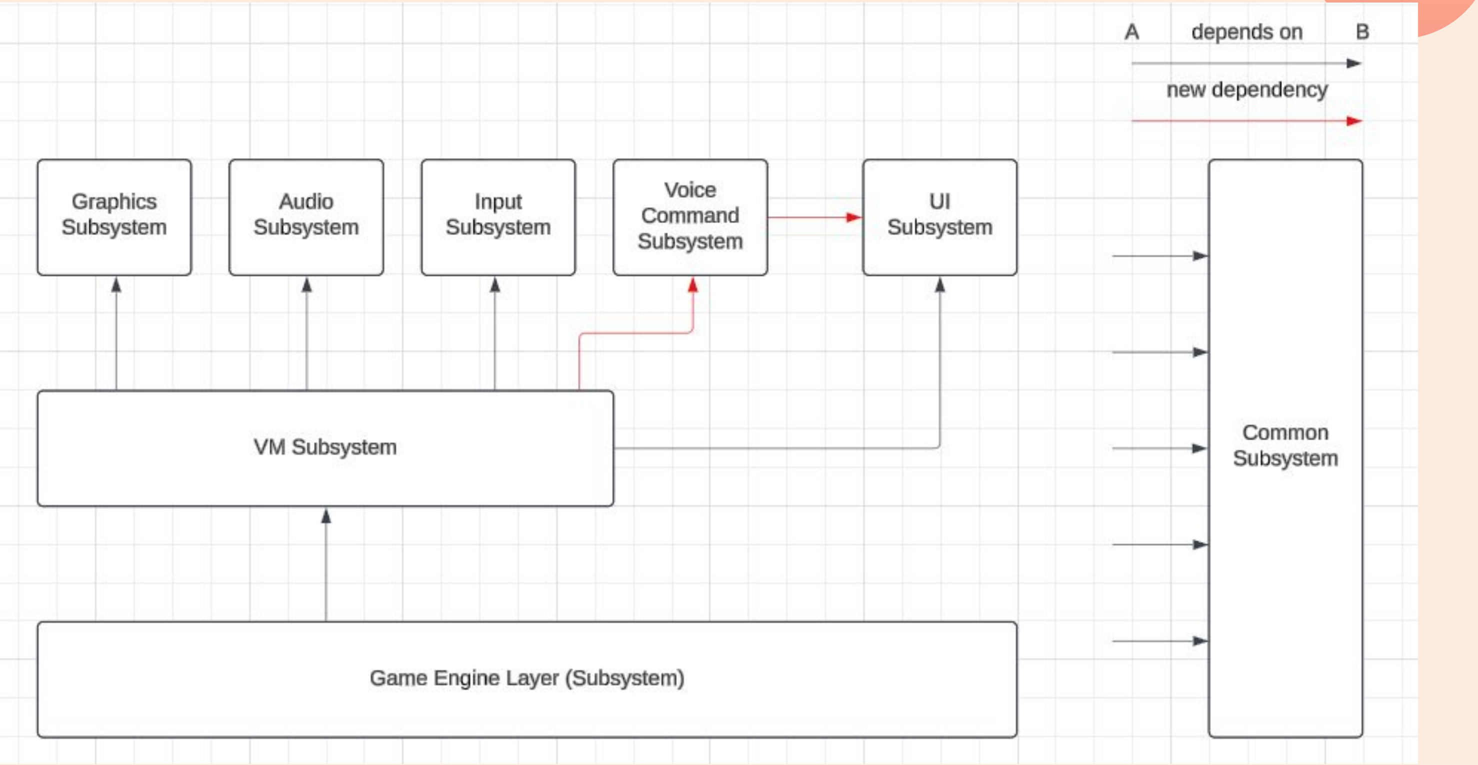


Figure 1. Architecture of ScummVM with the addition of the VCS as a standalone subsystem.

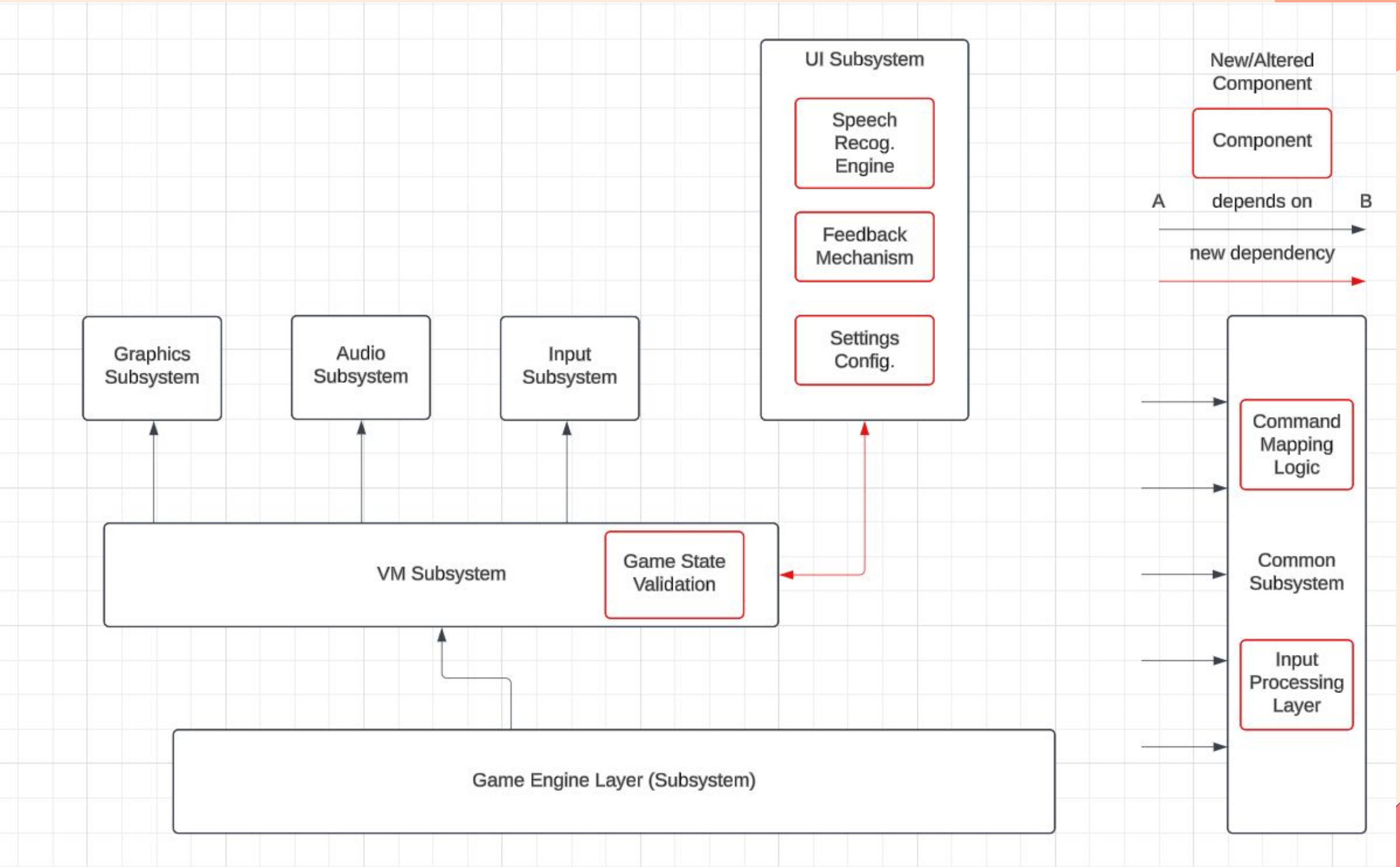


Figure 2. Architecture of ScummVM with the addition of the VCS as embedded system(s).

Comparison

Criteria	Standalone Subsystem	Embedded Approach
Usability	High	Moderate
Maintainability	High	Moderate
Scalability	Moderate	High
Testability	High	Low

Valid Command Use Case

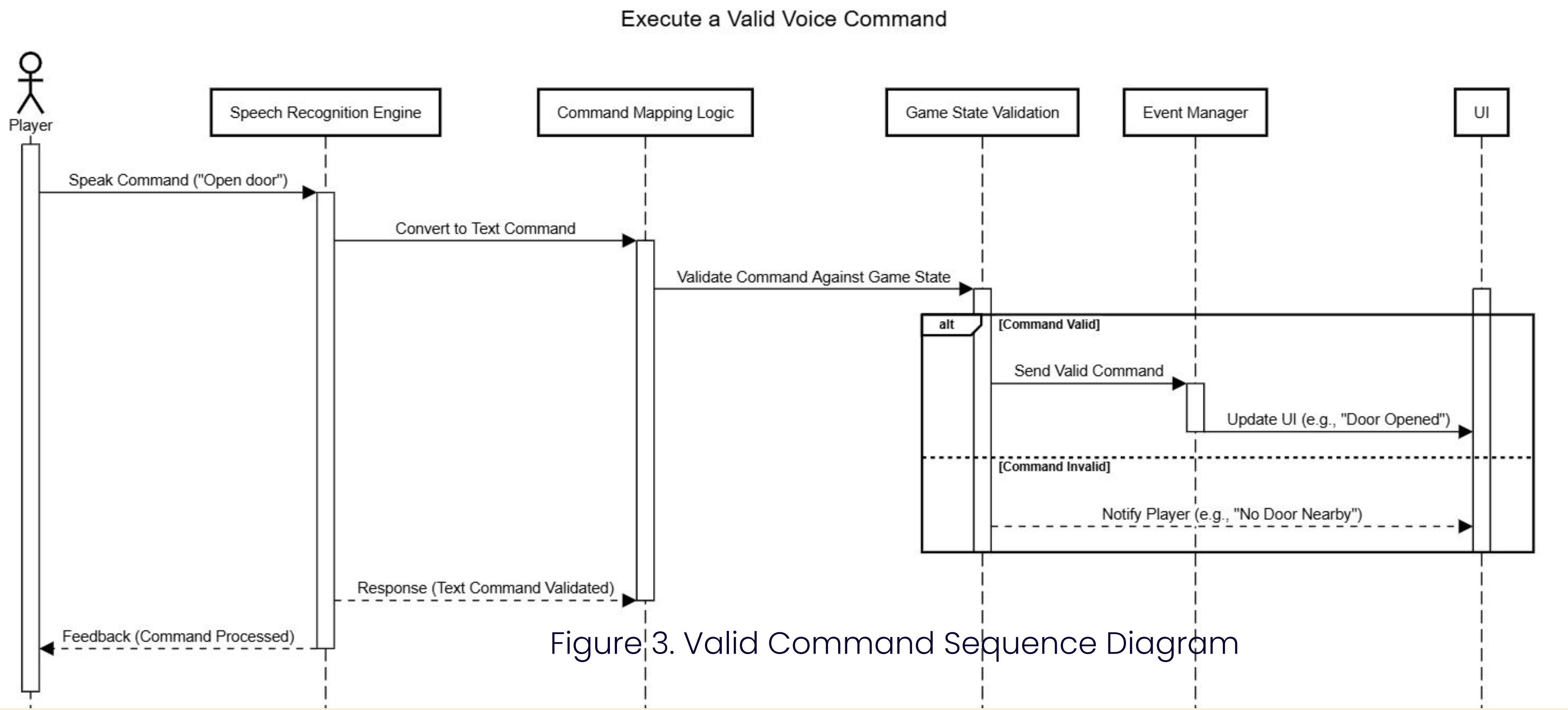

Execute a Valid Voice Command
Player
Speech Recognition Engine
Command Mapping Logic
Game State Validation
Event Manager
UI
Speak Command ("Open door")
Convert to Text Command
Validate Command Against Game State
alt
[Command Valid]
Send Valid Command
Update UI (e.g., "Door Opened")
[Command Invalid]
Notify Player (e.g., "No Door Nearby")
Response (Text Command Validated)
Feedback (Command Processed)

Figure 3. Valid Command Sequence Diagram

Invalid Command Use Case

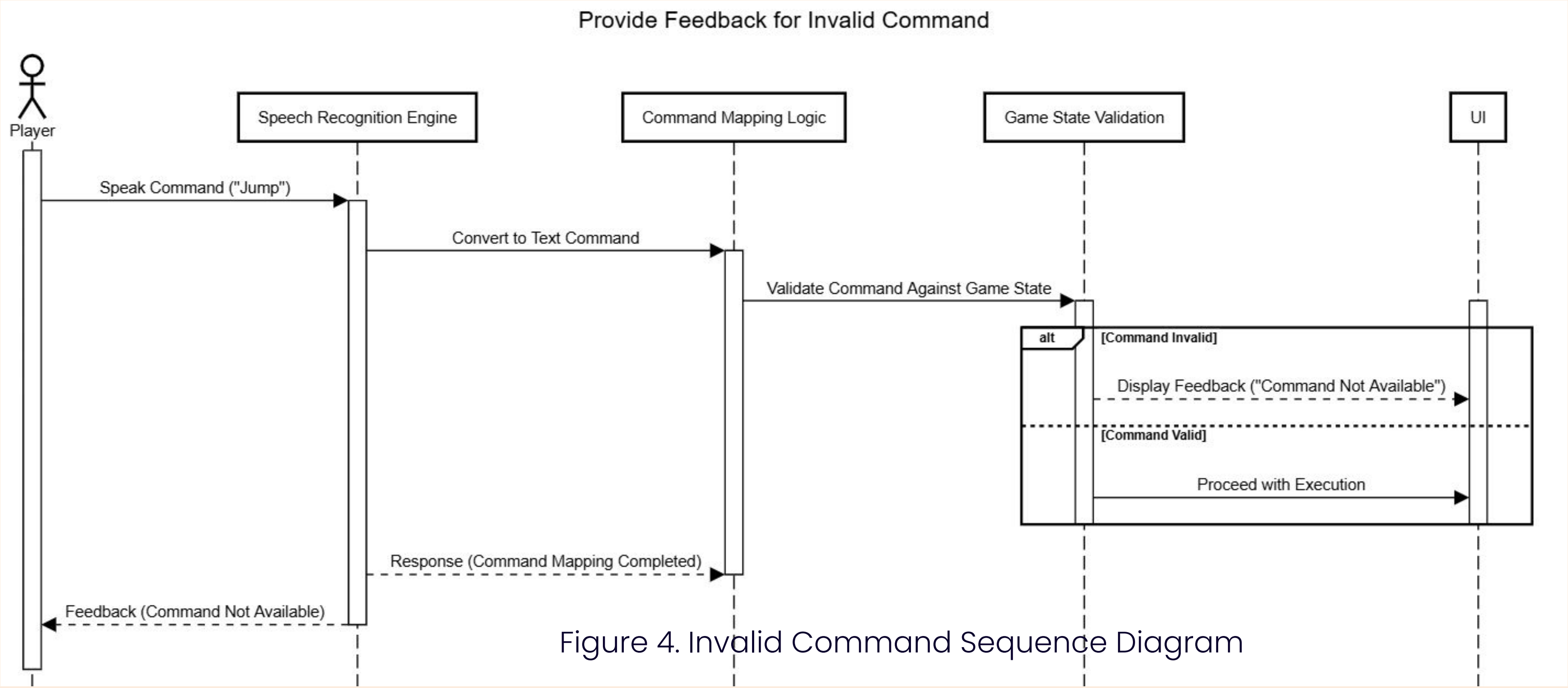

Provide Feedback for Invalid Command
Player
Speech Recognition Engine
Command Mapping Logic
Game State Validation
UI
Speak Command ("Jump")
Convert to Text Command
Validate Command Against Game State
alt
[Command Invalid]
Display Feedback ("Command Not Available")
[Command Valid]
Proceed with Execution
Response (Command Mapping Completed)
Feedback (Command Not Available)

Figure 4. Invalid Command Sequence Diagram

Risks & Limitations

1. Command Misrecognition
2. Integration Issues
3. Usability Concerns

Testing Plan

1. Usability Testing
2. Subsystem Compatibility/Integration Testing
3. Performance Testing

Concurrency

- **Mandatory as we cannot freeze the game when processing voice commands:**
  - Speech Recognition
  - Command Validation
  - Maintaining gameplay responsiveness

Lessons Learned

- Modularity is crucial for scalability and maintenance
- Isolating features reduces risk to existing functionality
- Concurrency is a must-have for responsiveness
- Comprehensive testing, specifically integration ensures reliability

Conclusion

Evaluated two approaches:

- Standalone subsystem: isolates voice functionality.
- Embedded implementation: integrates into existing subsystems.
- Recommendation: Standalone subsystem approach.

Advantages:

  - Greater modularity, maintainability, and testability.
  - Ensures smoother integration with fewer disruptions.
  - Enhances reliability with asynchronous processing.
  - Preserves ScummVM's core functionality.
  - Supports future accessibility improvements.

Thank you