CISC 322/326

A2 Concrete Architecture Presentation.

Youtube Link: https://youtu.be/piqyRPxp7AM

Group Roles

Group Name:

MAWLOK

Team Lead:

Owen Meima

Presenters:

Andrew Zhang
Michael Marchello

Other Team Members:

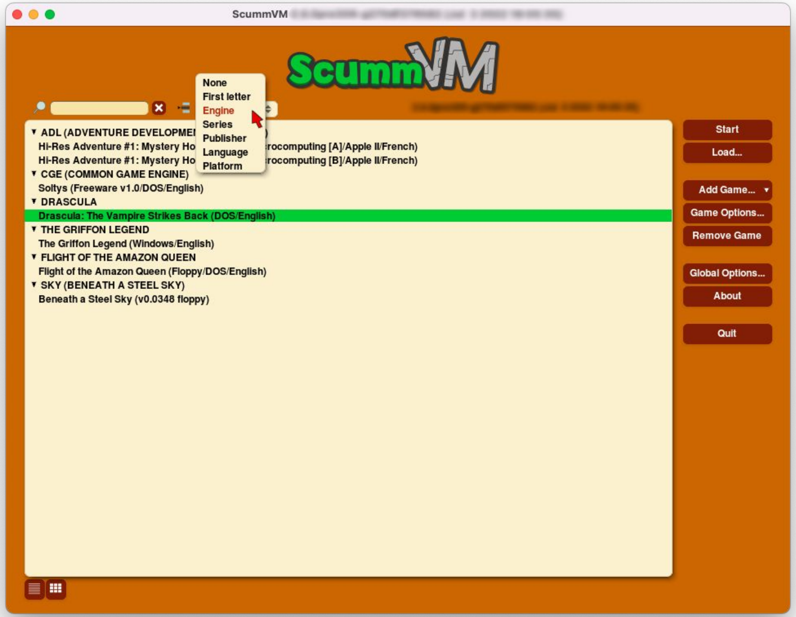
Lance Lei
Kevin Panchalingam
Zhuweinuo Zheng

Introduction

- Re-analyze Scummvm Software using the Understand IDE tool. To find unexpected dependencies.

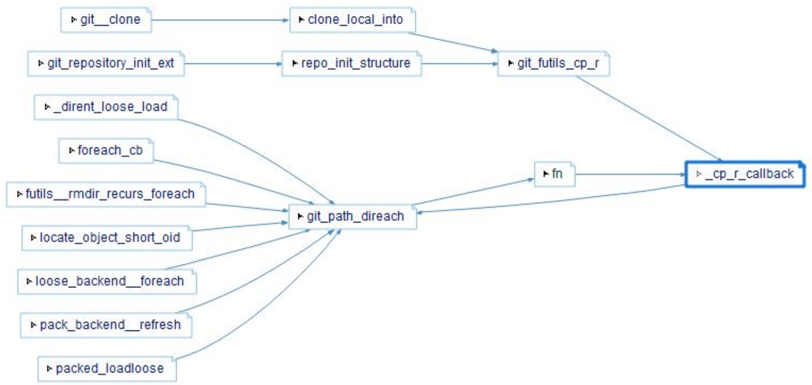
- This includes absences, convergences, and divergences.

- Utilize this knowledge to modify our concrete and conceptual architecture.



▼ 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)

Understand

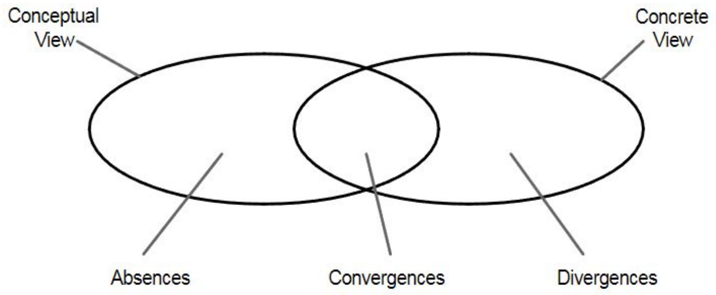


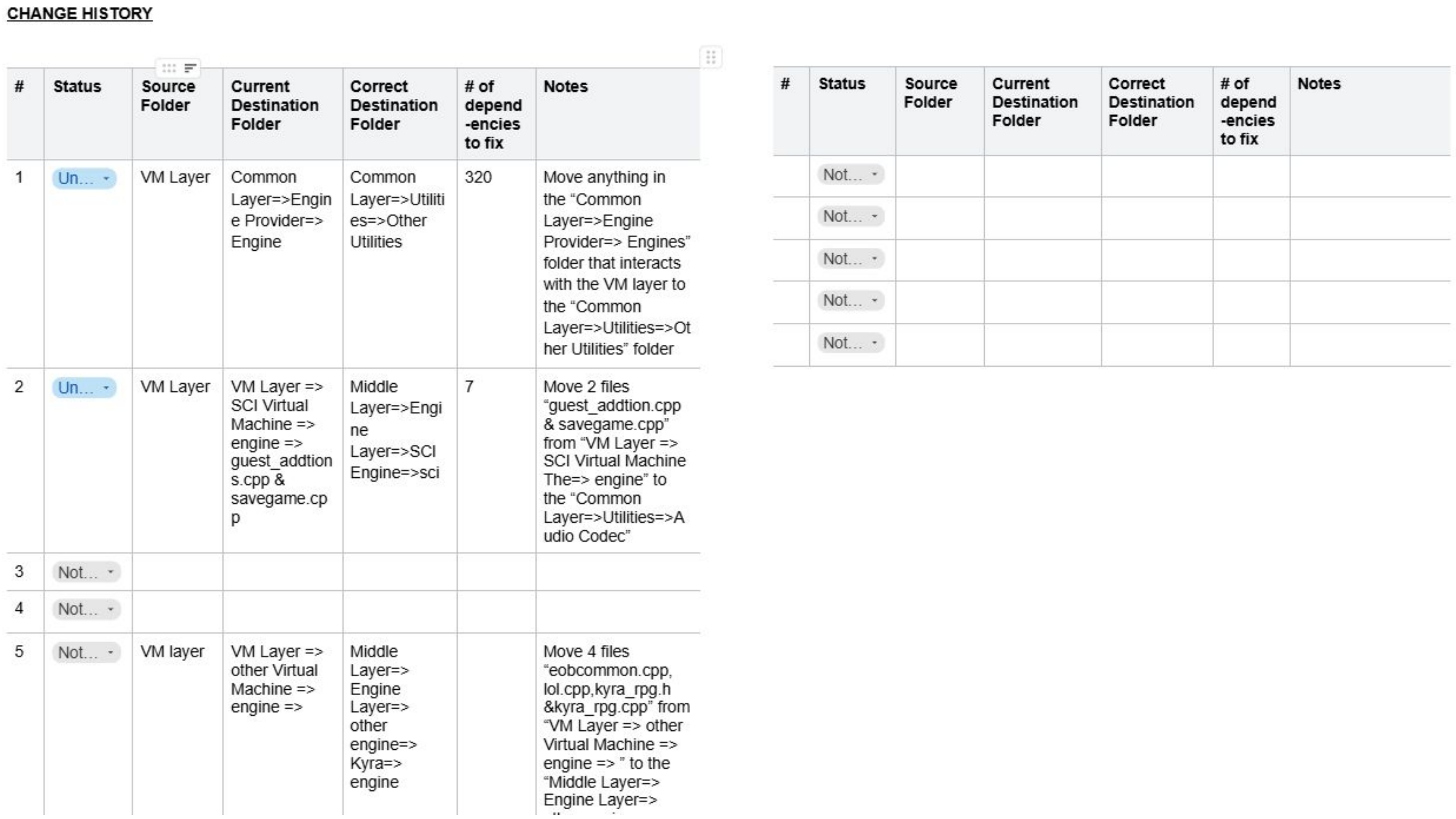
Understand™ by SciTools

- Understand is a software that enables static code analysis. With tools such as an array of visuals, documentation, and metric tools to aid the user.
- Was made to aid software developers comprehend, maintain, and document their source code.

Derivation Process

- Used Understand to view the concrete architecture.
- Looked at components and files (including code) to determine absences, divergences, and convergences.
- If changes were made to the concrete architecture, they were documented in a chart (see next slide).





CHANGE HISTORY

#	Status	Source Folder	Current Destination Folder	Correct Destination Folder	# of depend -encies to fix	Notes
1	Un…	VM Layer	Common Layer=>Engine Provider=> Engine	Common Layer=>Utilities=>Other Utilities	320	Move anything in the "Common Layer=>Engine Provider=> Engines" folder that interacts with the VM layer to the "Common Layer=>Utilities=>Other Utilities" folder
2	Un…	VM Layer	VM Layer => SCI Virtual Machine => engine => guest_addtions.cpp & savegame.cpp	Middle Layer=>Engine Layer=>SCI Engine=>sci	7	Move 2 files "guest_addtion.cpp & savegame.cpp" from "VM Layer => SCI Virtual Machine The=> engine" to the "Common Layer=>Utilities=>Audio Codec"
3	Not…					
4	Not…					
5	Not…	VM layer	VM Layer => other Virtual Machine => engine =>	Middle Layer=> Engine Layer=> other engine=> Kyra=> engine		Move 4 files "eobcommon.cpp, lol.cpp,kyra_rpg.h &kyra_rpg.cpp" from "VM Layer => other Virtual Machine => engine => " to the "Middle Layer=> Engine Layer=>

#	Status	Source Folder	Current Destination Folder	Correct Destination Folder	# of depend -encies to fix	Notes
	Not…					
	Not…					
	Not…					
	Not…					
	Not…					

Derivation Process (Chart)

Old

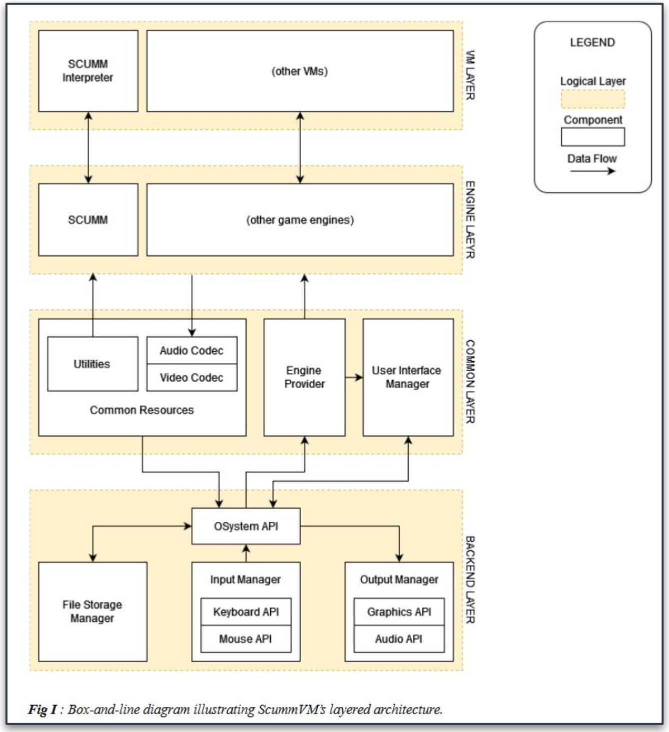
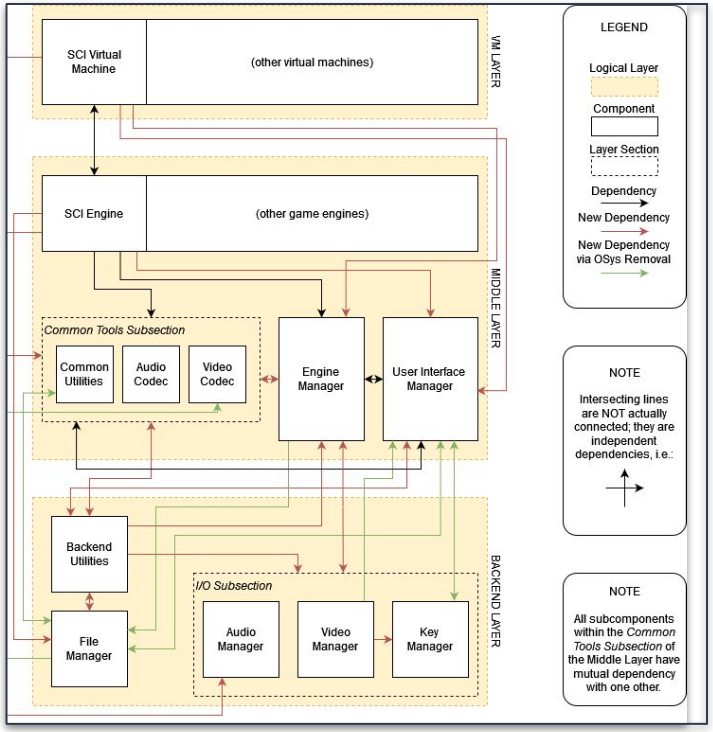
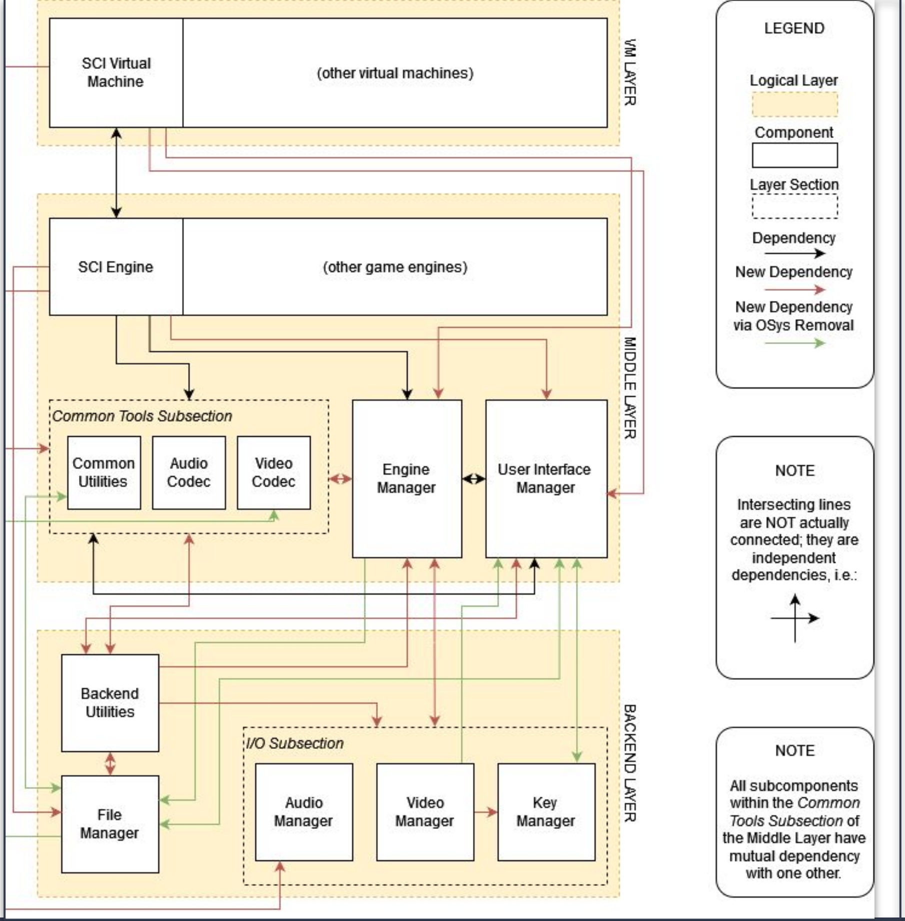


Fig I : Box-and-line diagram illustrating ScummVM's layered architecture.

New

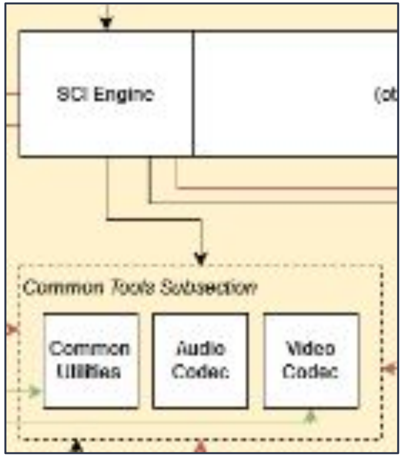


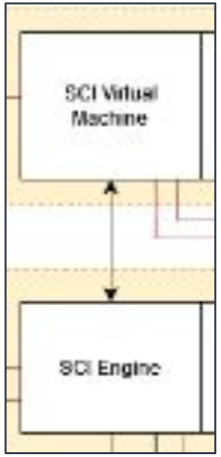
Architectural Overview (Conceptual)

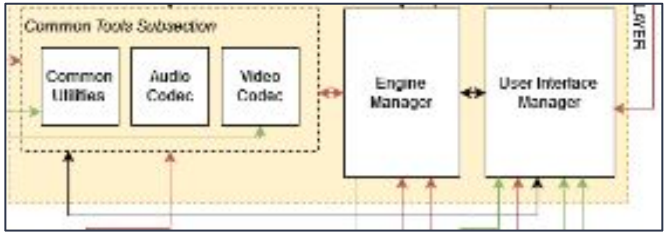


Architectural Overview (Concrete)

Convergences





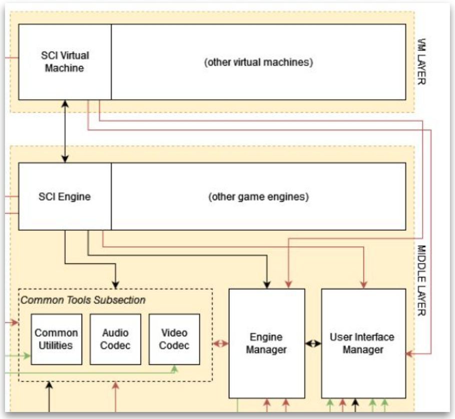


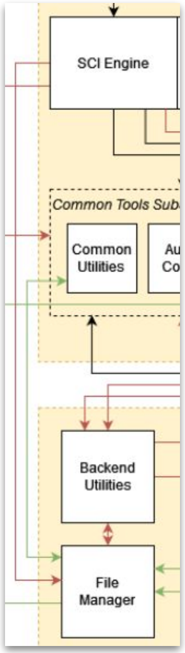
- The SCI Engine (formerly Scumm) relies on the common tools section for shared resources like audio, video, and utilities, and depends on the engine manager for managing files and classes.
- The virtual machine components and game engine components remain interdependent, enabling gameplay on the user's computer.
- Communication persists between the common tools subsection, user interface manager, and engine manager, allowing players to adjust game settings and engine preferences.

Divergences

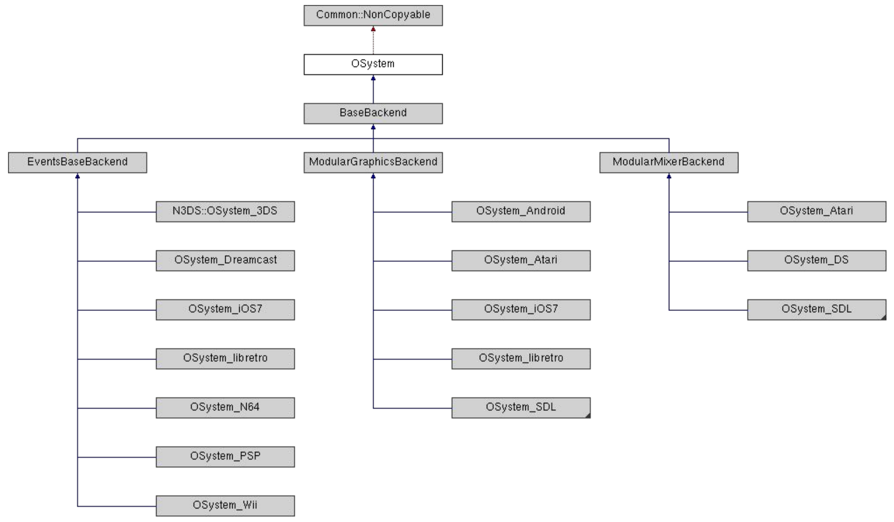
- There were multiple divergences found in the the system, highlighted by red arrows. While the green arrows are formed after removing the OSystem API

- Examples:
  - The Engine manager and the User Interface manager are now dependent on the SCI virtual machine component, whereas initially, only the Sci engine was dependent on it.
  - The file manager within the backend layer is now directly dependent on the SCI engine component.





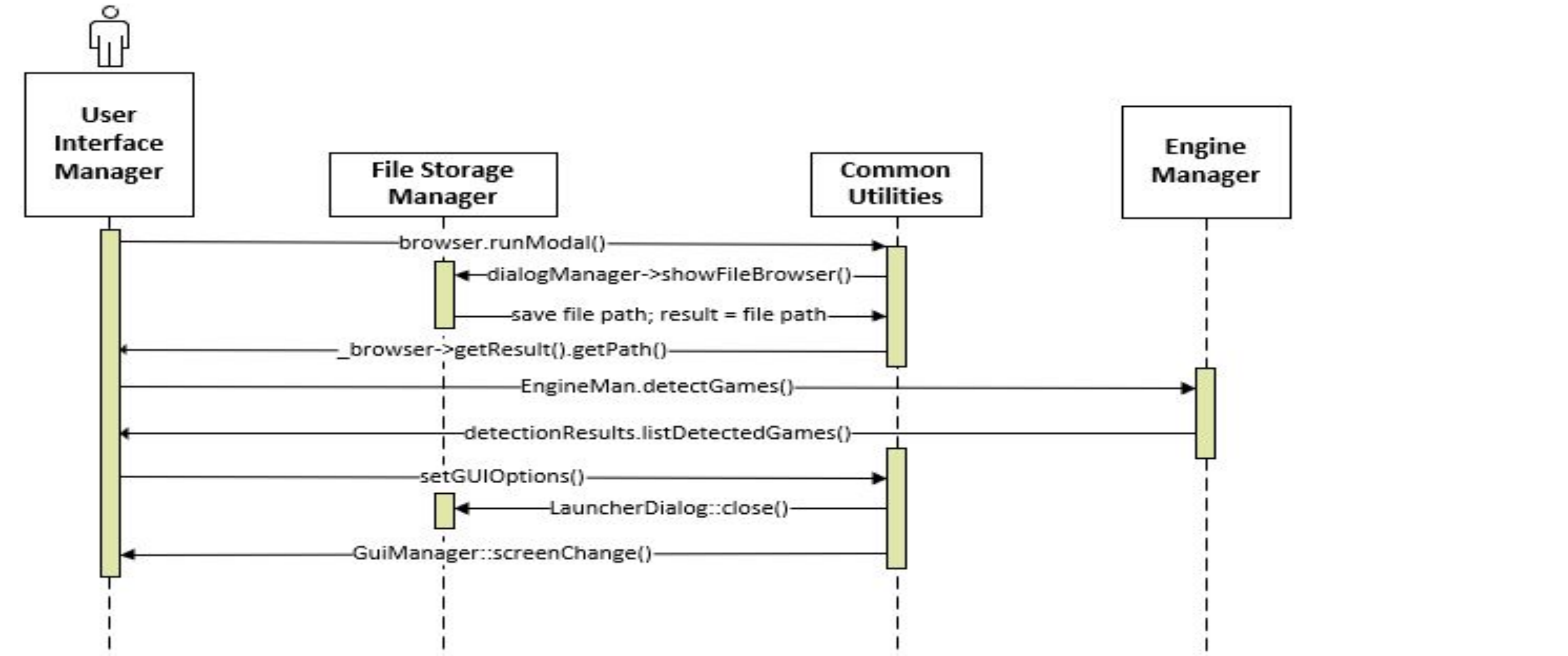
Absences



- The Osystem API was removed because it functioned as a super class for the entire backend layer, serving as an interface for various backends like scummVM distribution.
- Dependencies previously relying on the Osystem API now connect directly to their respective components, such as the Key Manager accessing the User Interface Manager without needing the Osystem API.

Sequence Diagrams (Use Case 1)

Use-Case 1: Player loads the data files for a new valid game they haven't played



Lessons Learned

- How Understand works and how to use it.
- The process of finding convergences, divergences, and absences using a conceptual and concrete architecture.
- There’s value proper documentation of a system.

Conclusion

- The concrete architecture of the system has been addressed, along with hidden dependencies discovered.
- Using Understand, multiple divergences and minor absences in the system's components were identified.
- The conceptual architecture had to be reevaluated to align with the unexpected dependencies, leading to the merging of certain layers.
- Despite the complexity of the process, the architecture was carefully designed for user-friendly functionality.