Team Torrent (Group 25)

CISC 322/326 Assignment 2: Presentation

Kodi: Concrete Architecture Analysis

Aselstyne, Alex (alex.aselstyne@queensu.ca) Lead.

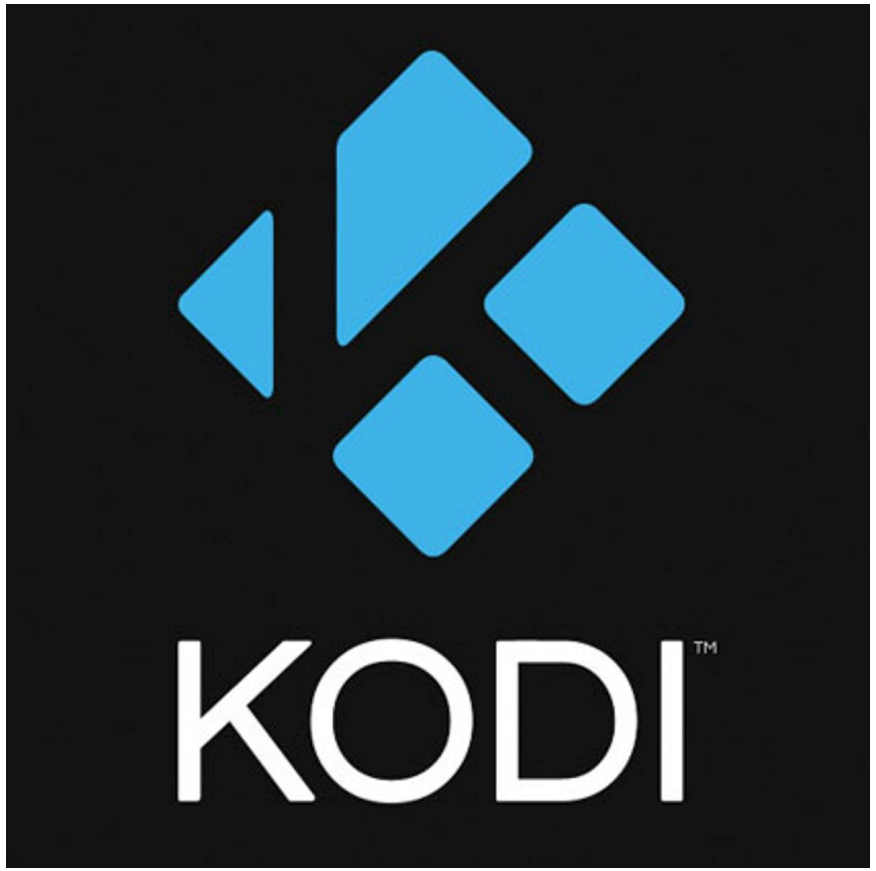
Dinari, Daniel (20dd29@queensu.ca) Pres.

Nagel, Jake (20jn29@queensu.ca)

Peterson, Jack (21jrp10@queensu.ca) Pres.

Pleava, Ryan (20rcp5@queensu.ca)

YouTube Link: https://youtu.be/Ae_2iAKzyM0



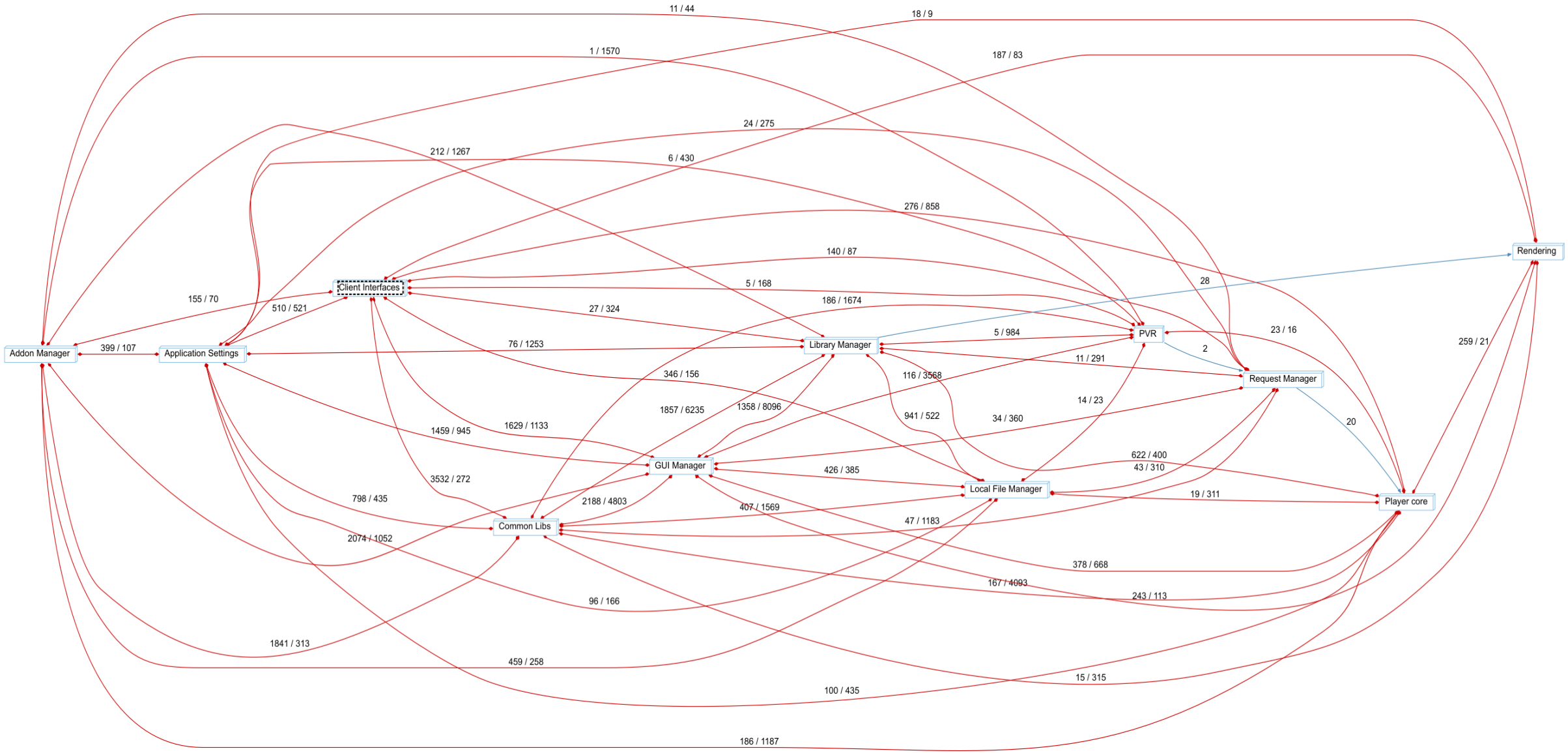
Introduction to Kodi

- Free and open-source multimedia player.
- Originally developed for the Xbox (2001), initial release in 2003.
- Later ported to most popular platforms.
- Disassociated from Xbox in 2014.

Introduction to our Project

- Document Kodi's conceptual architecture abstractly
- 5 Primary Topics
    1) Overview of Top Level Concrete Subsystems
    2) Derivation Process
    3) Describing the Use Cases and Sequence Diagrams
    4) Reflection Analysis – Player Core Subsystem
    5) Reflection Analysis – High Level Architecture
    6) Conclusions

Overview of Top Level Concrete Subsystems



Derivation Process

Utilized Understand, a tool for viewing repositories.

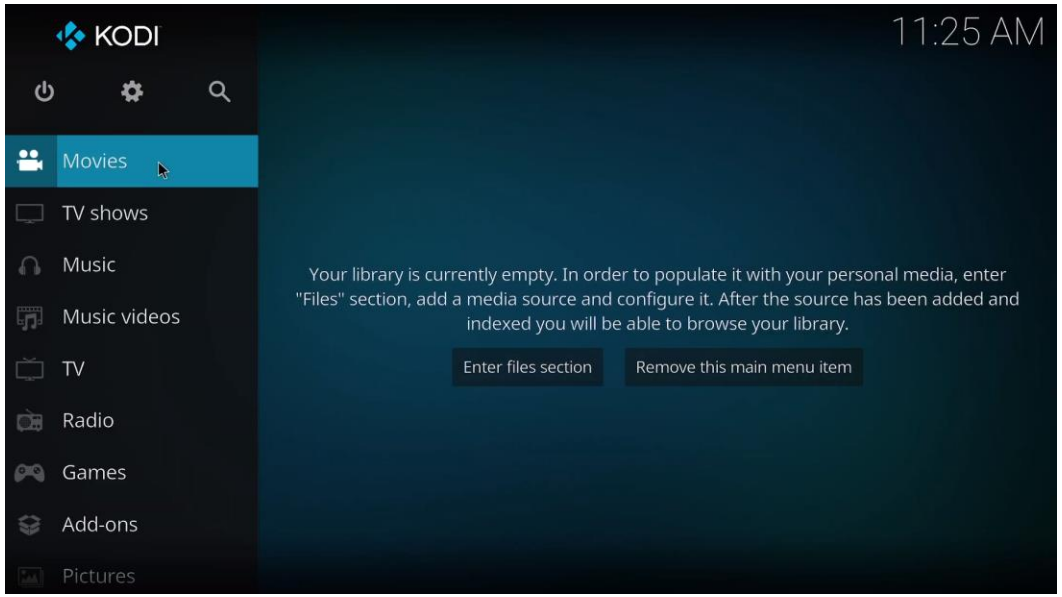
Started with our original conceptual architecture as a starting point.

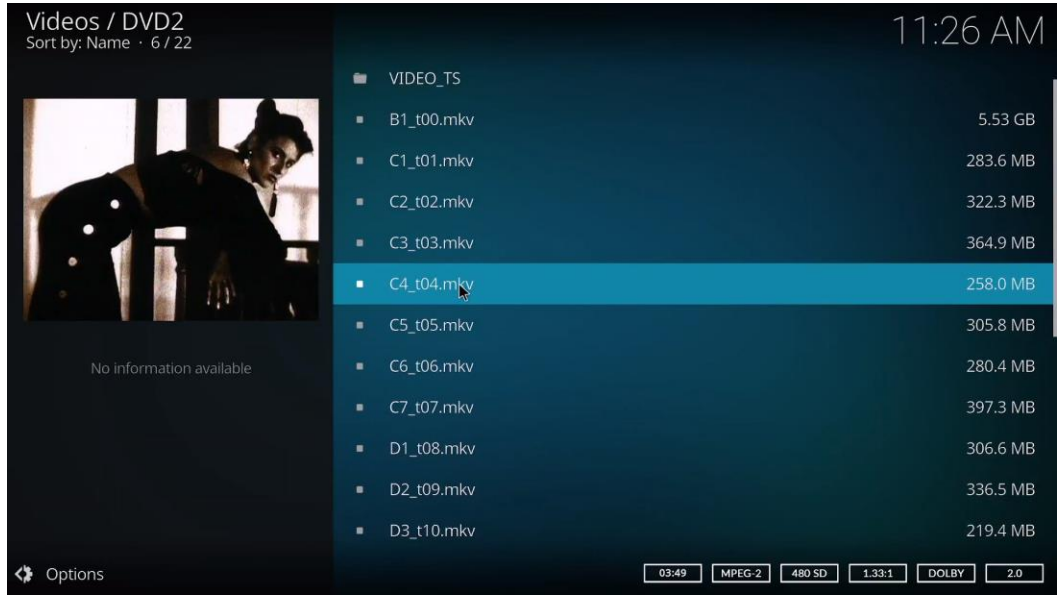
Mapped files to components, creating new components or altering names as necessary.

Dug through code to generate new sequence diagrams, and for reflection analysis.

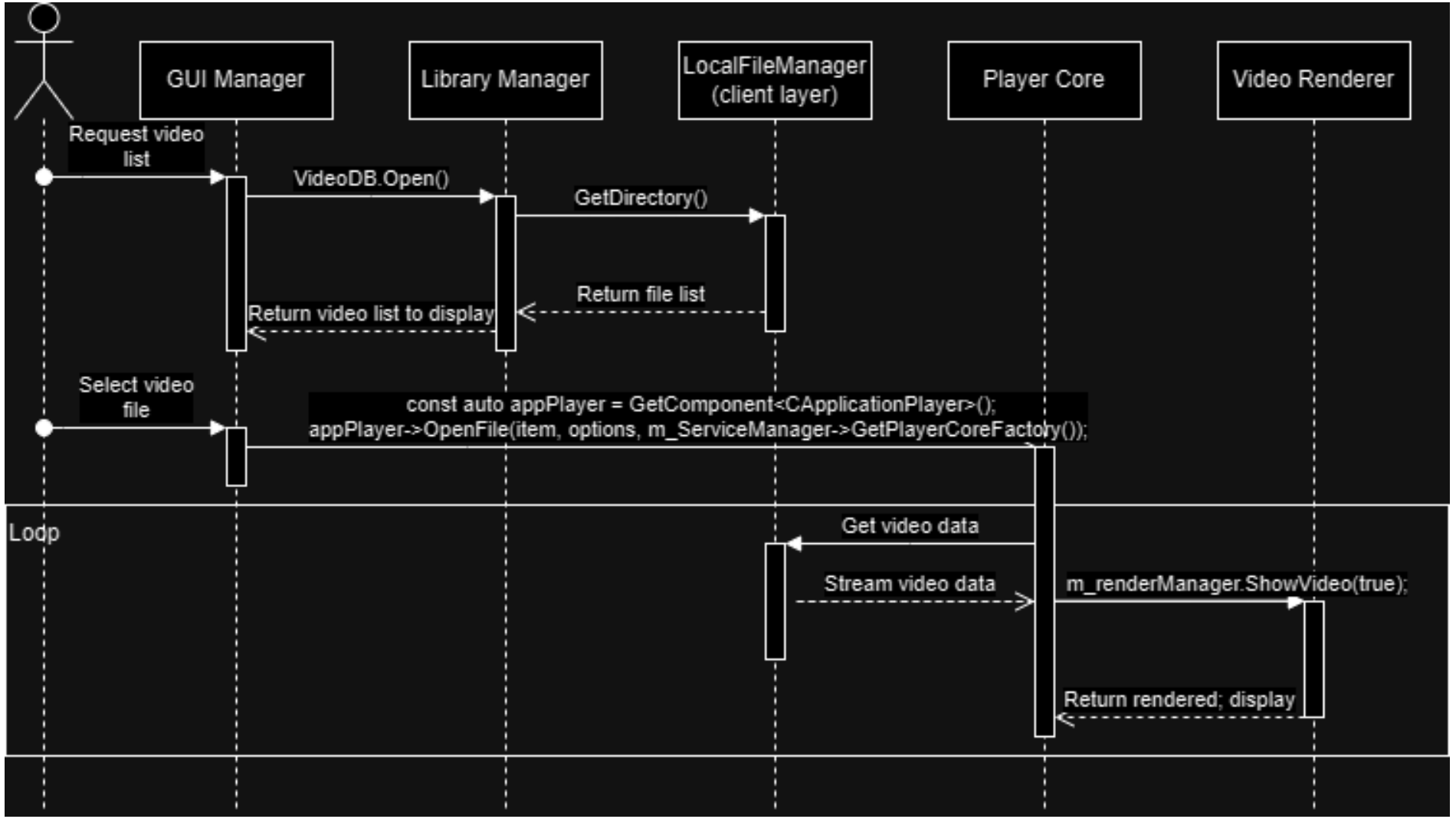
Use Case 1

- User selecting and playing a video





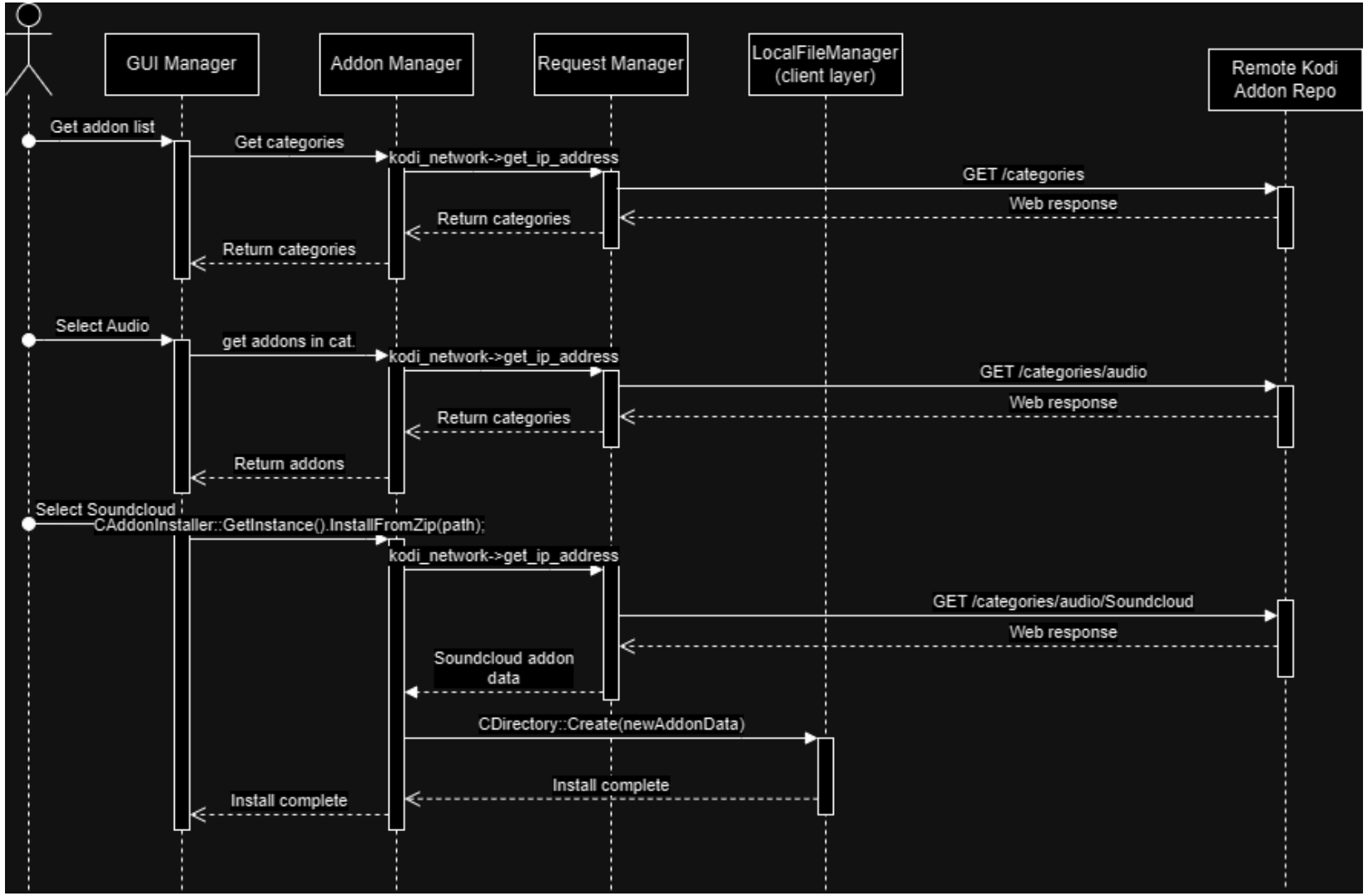
Use Case 1 Sequence Diagram



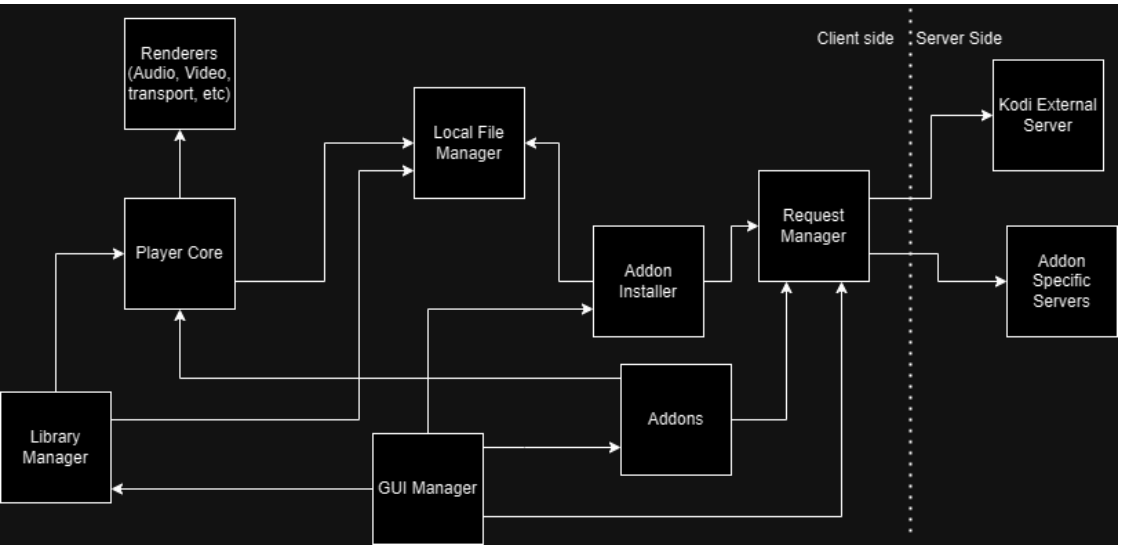
Use Case 2

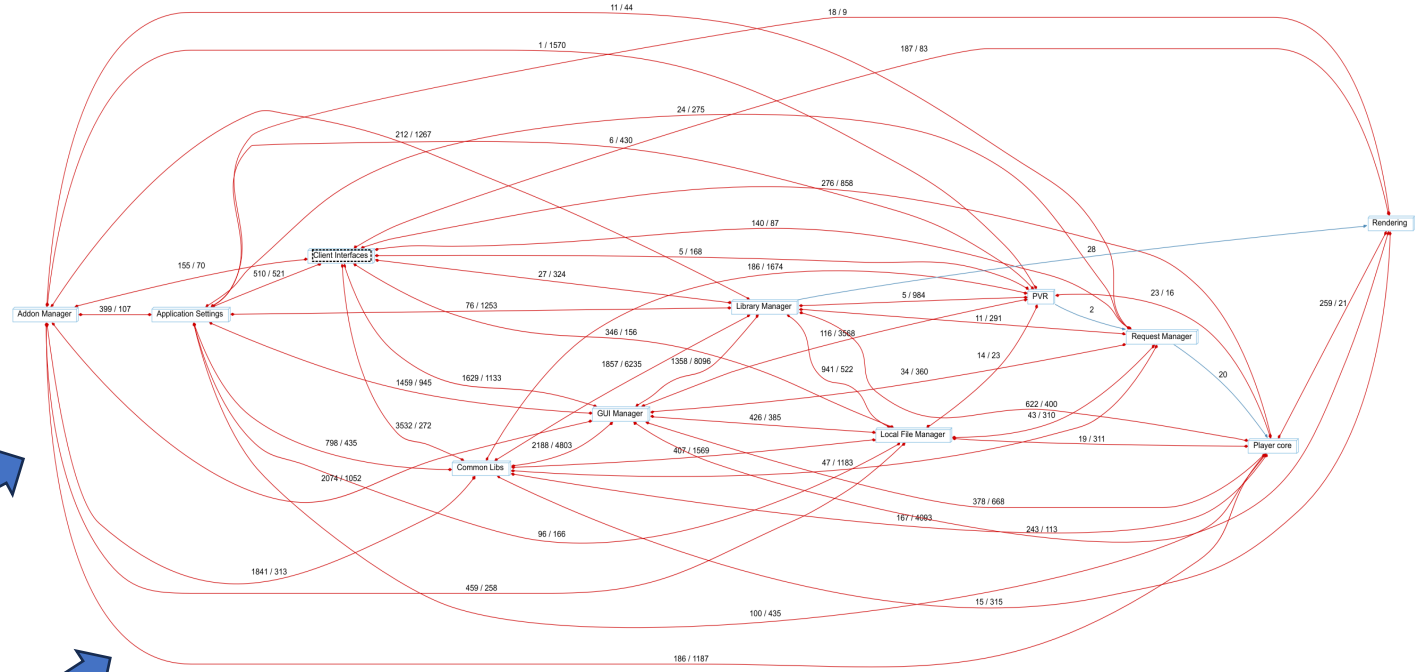
- User selects an addon from the available list, installs it, and then uses it to play a song from a remote server

Sequence Diagrams for Use Case 2



Reflection Analysis of High Level Architecture



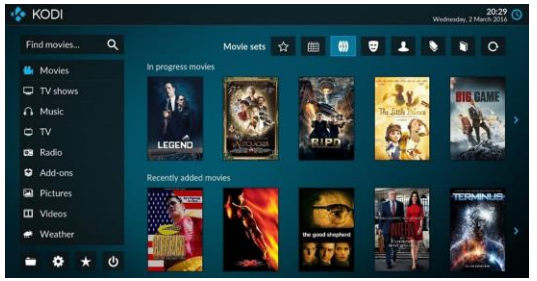


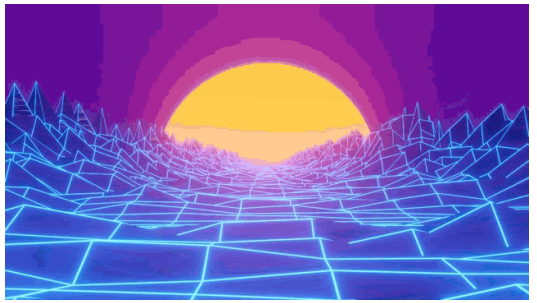
Reflection Analysis of Player Core

VideoPlayer

AudioPlayer

RetroPlayer





Lessons Learned and Conclusion

- Kodi is a highly interconnected system, many dependencies.
- Large-scale software will almost never be implemented in the exact way it was conceptually designed.
- Repository was much larger than anticipated, working as a group to distribute work was important.
- Kodi is a very large software, likely that there are some misunderstands of the architecture we didn't catch.
- Lots of dependencies are difficult to figure out with only conceptual architecture, due to intricacies in the code.
- Kodi is a very versatile and well-designed software, it recycles a lot of code and is efficient.

References

[1] “About Kodi,” Kodi.tv. [Online]. Available: https://kodi.tv/about/. [Accessed: 22-Oct-2023].

[2] Kodi.wiki. [Online]. Available: https://kodi.wiki/view/Architecture#Business_Layer.[Accessed: 22-Oct-2023].

[3] “Kodi,” Github.io. [Online]. Available: http://delftswa.github.io/chapters/kodi/. [Accessed: 22-Oct-2023].

[4] “Kodi Foundation,” Kodi.tv. [Online]. Available: https://kodi.tv/about/foundation/. [Accessed: 22-Oct-2023].

[5] Kodi.wiki. [Online]. Available: https://kodi.wiki/view/History_of_Kodi. [Accessed: 22-Oct-2023].

[6] “Pipe and filter,” Berkeley.edu. [Online]. Available: https://patterns.eecs.berkeley.edu/?page_id=19. [Accessed: 22-Oct-2023].