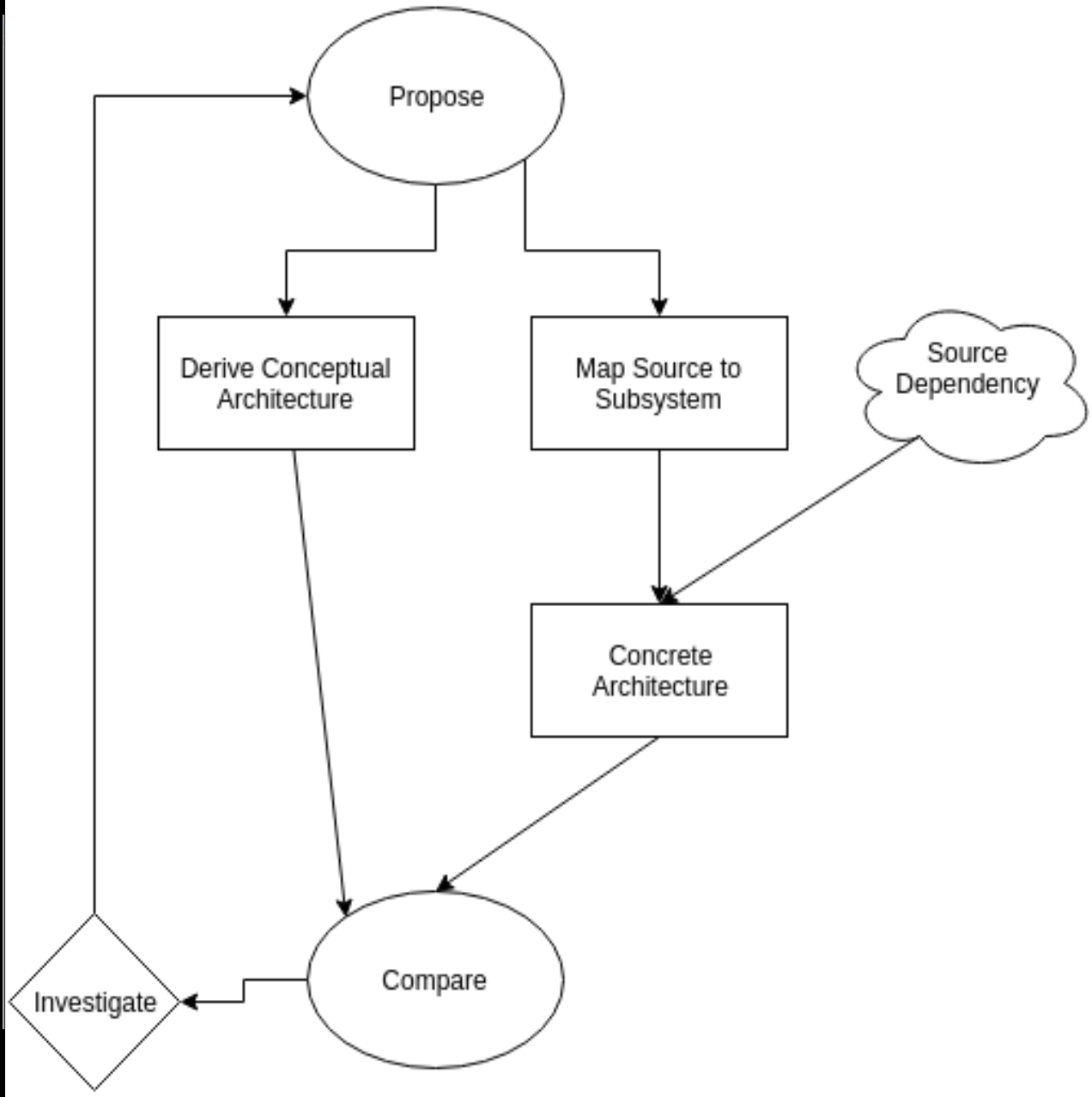
Pangolin: Concrete Architecture of Super Tux Kart

A CISC 326 Project by:
Mohammed Gasmallah
Russell Dawes
Caleb Aikens
Leonard Ha
Vincent Hung
Joseph Landy

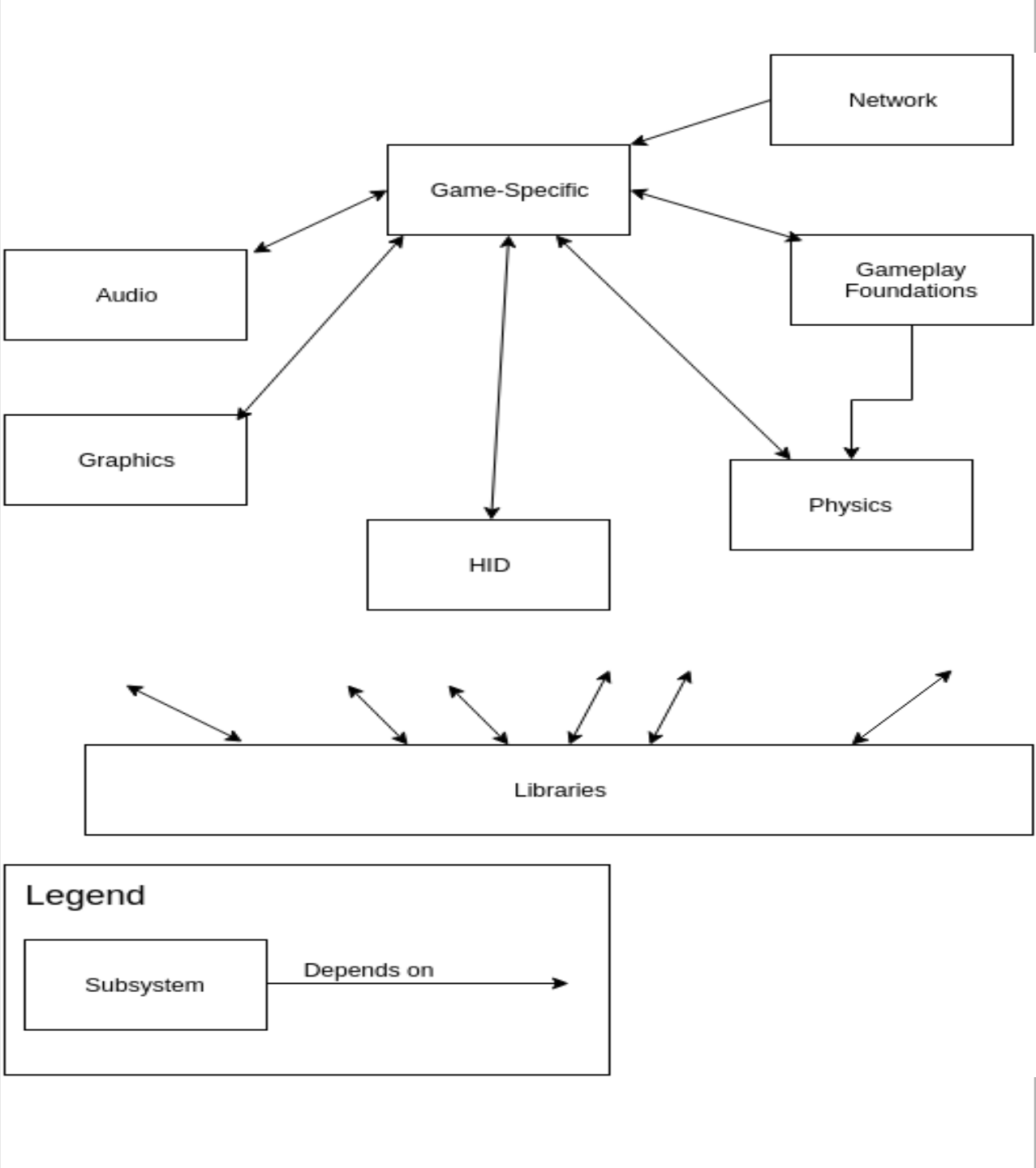
Overview

- Derivation Process
- Revised Conceptual Architecture
- Concrete Architecture
- Dependencies Analysis
- Architectural Styles
- Sequence Diagrams
- Concurrency
- Limitations & difficulties
- Lessons Learned
- Conclusion

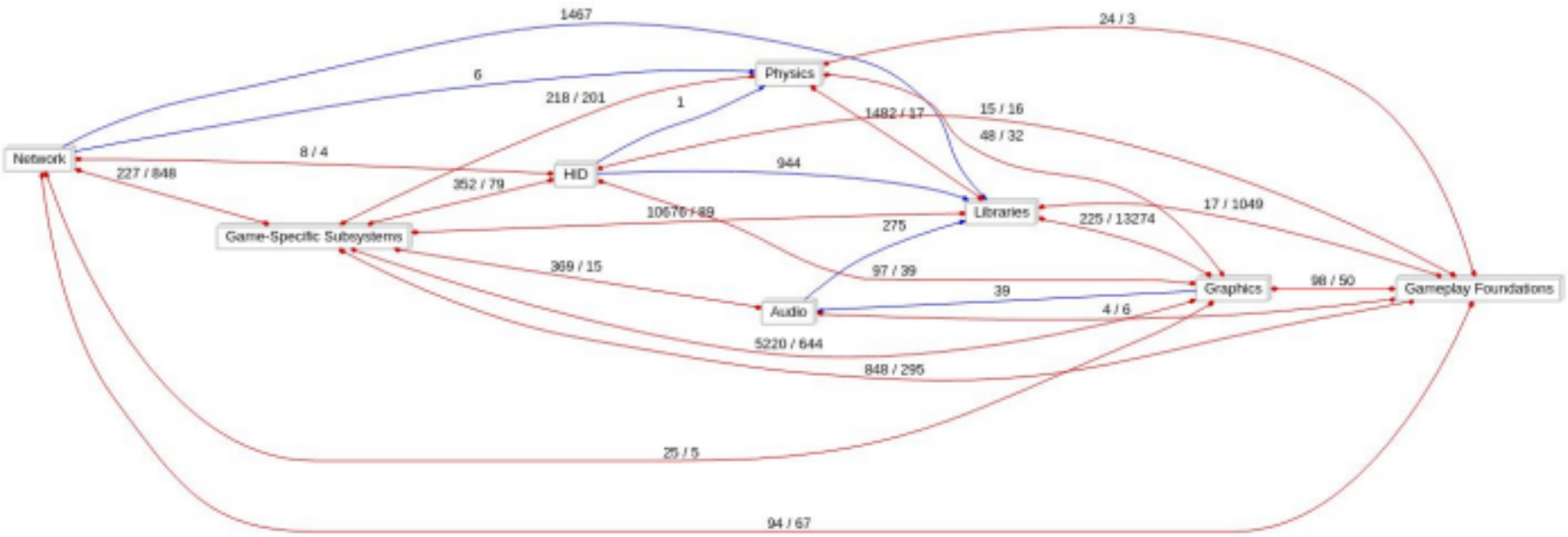
Derivation Process



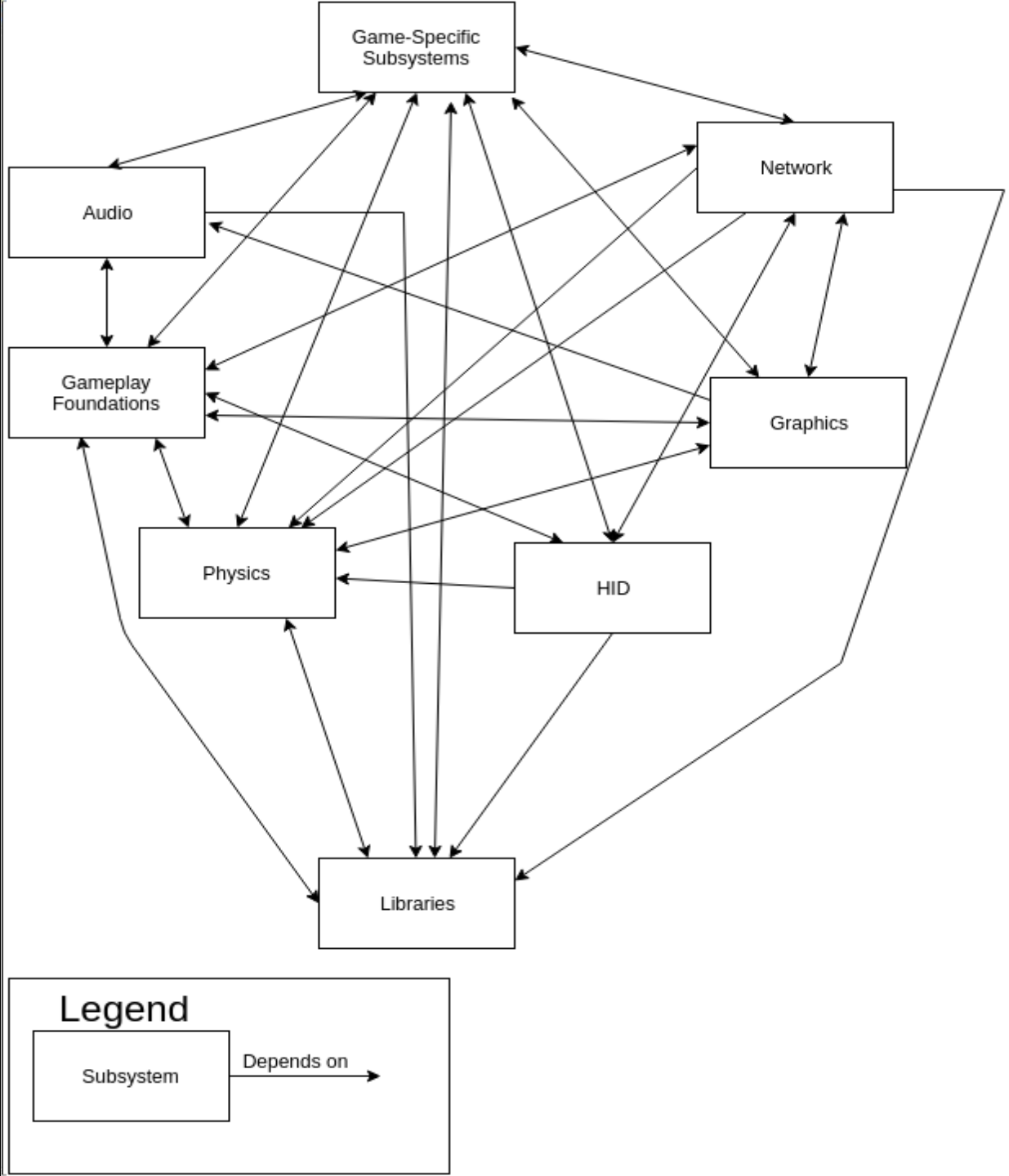
Revised Conceptual Architecture



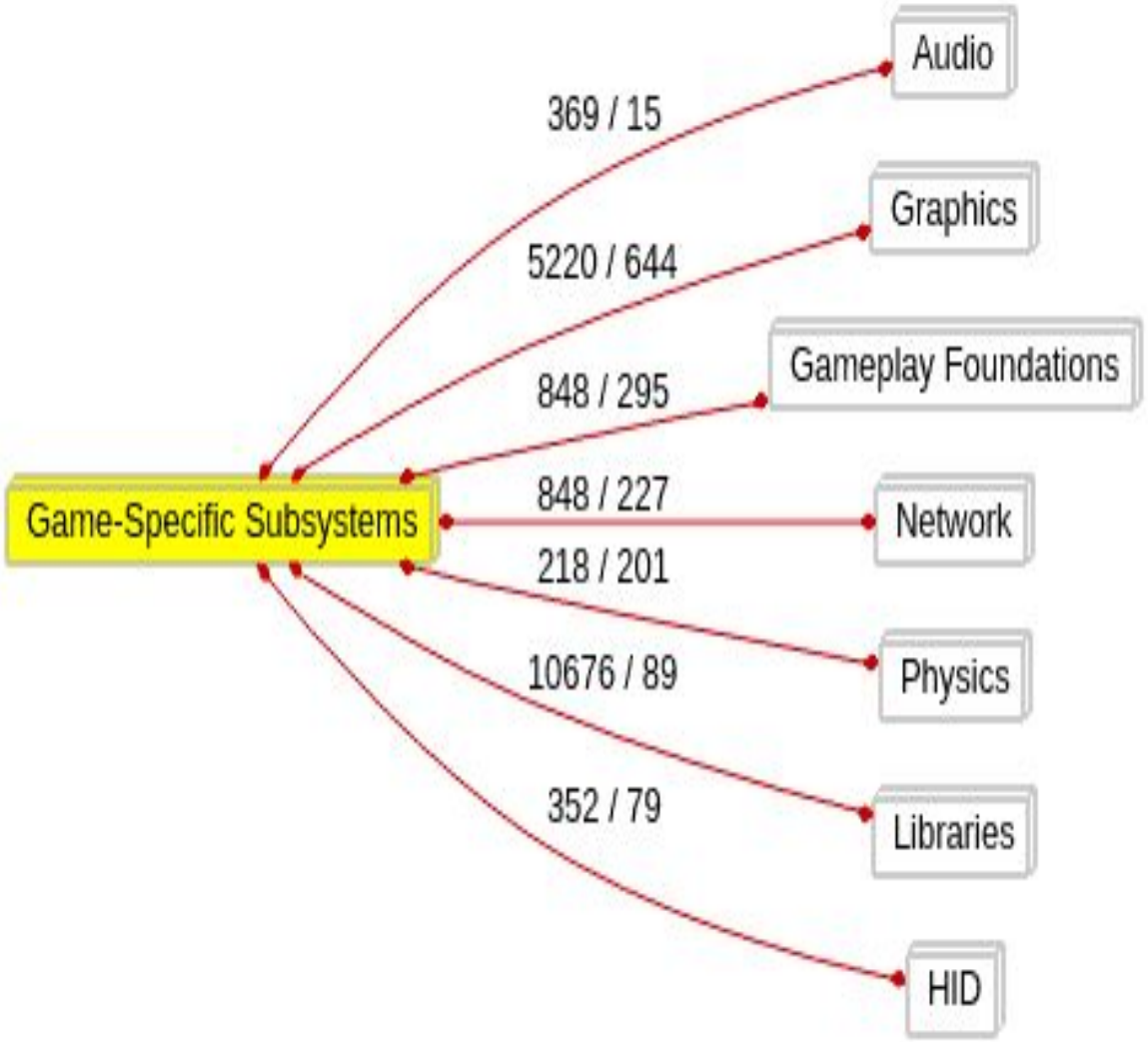
Concrete Architecture: From Understand



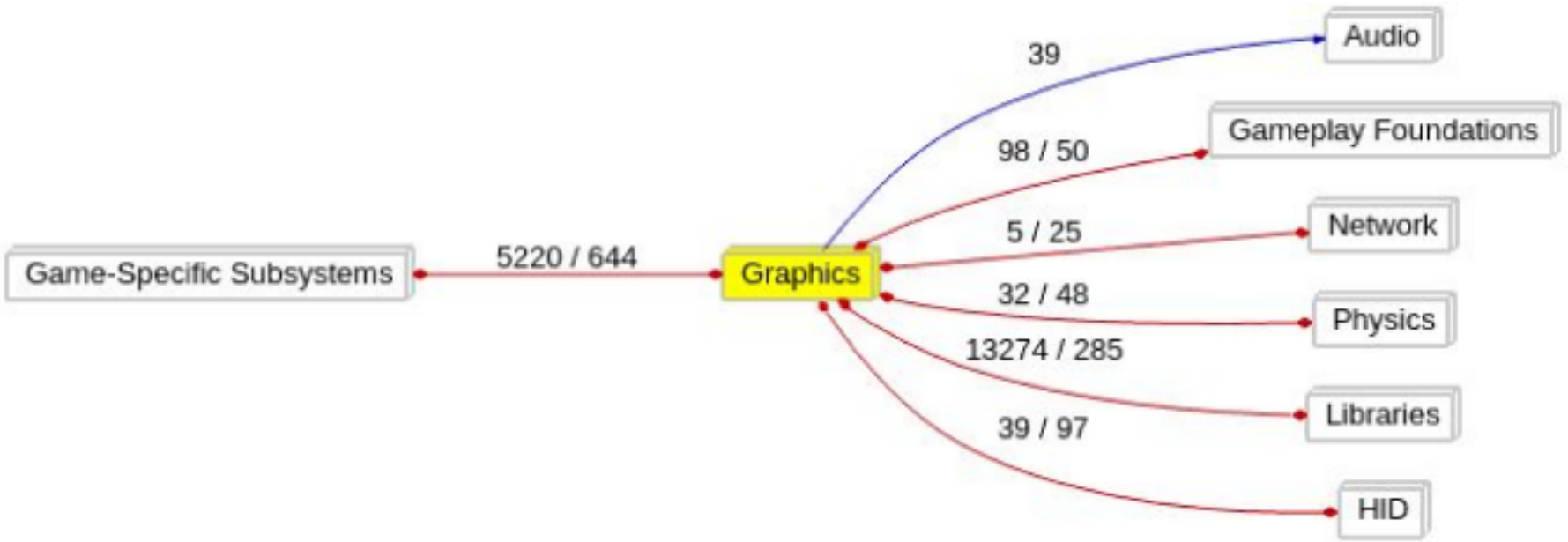
Concrete Architecture: Simplified



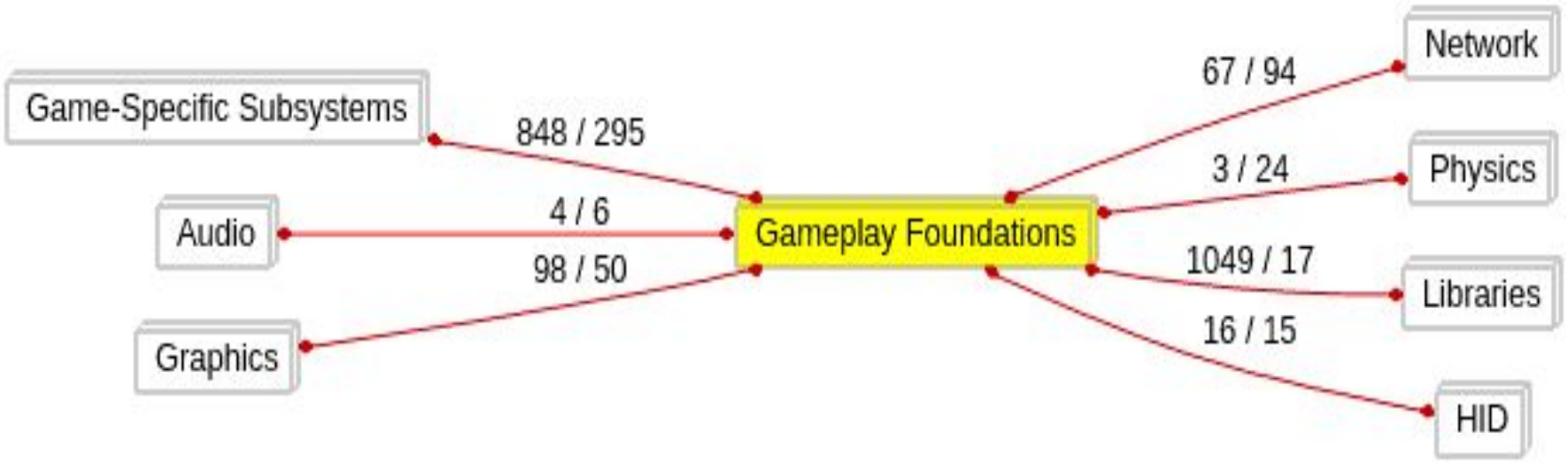
Dependencies: Game-Specific Subsystems



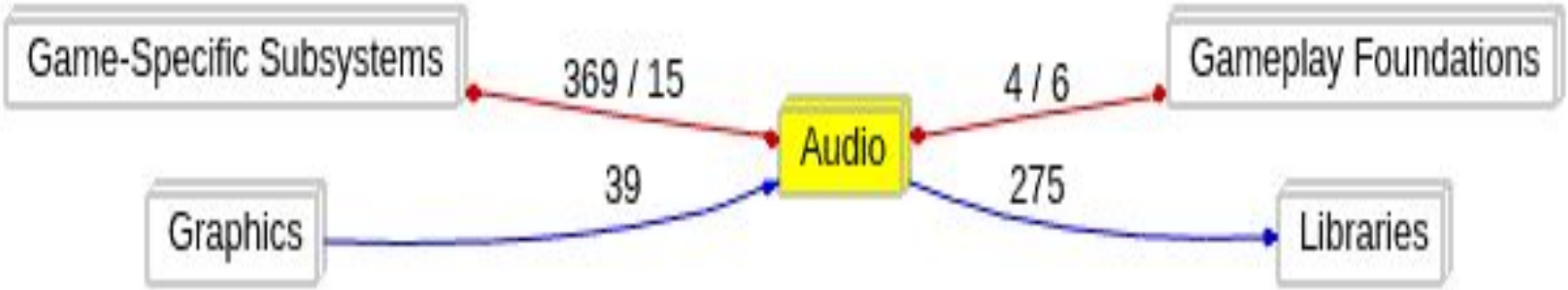
Dependencies: Graphics



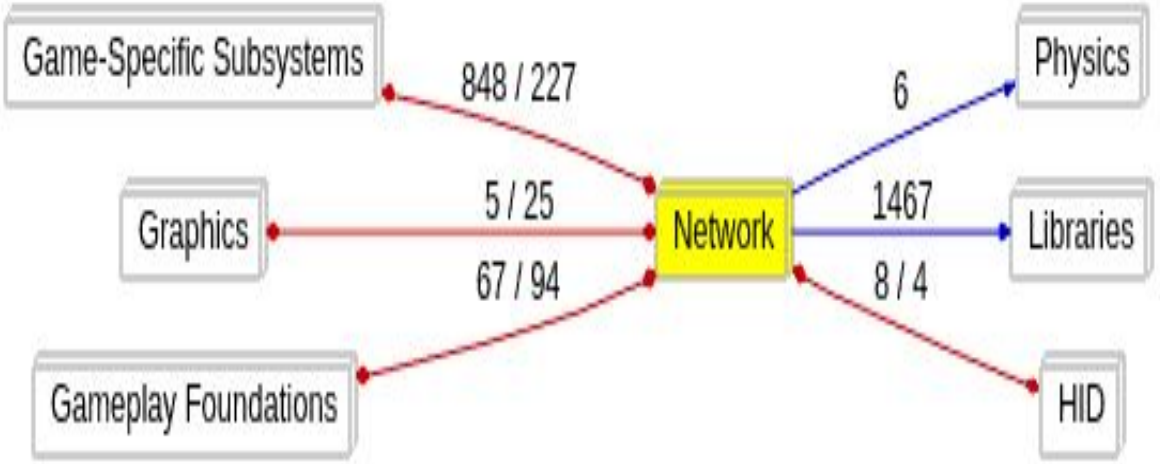
Dependencies: Gameplay Foundations



Dependencies: Audio

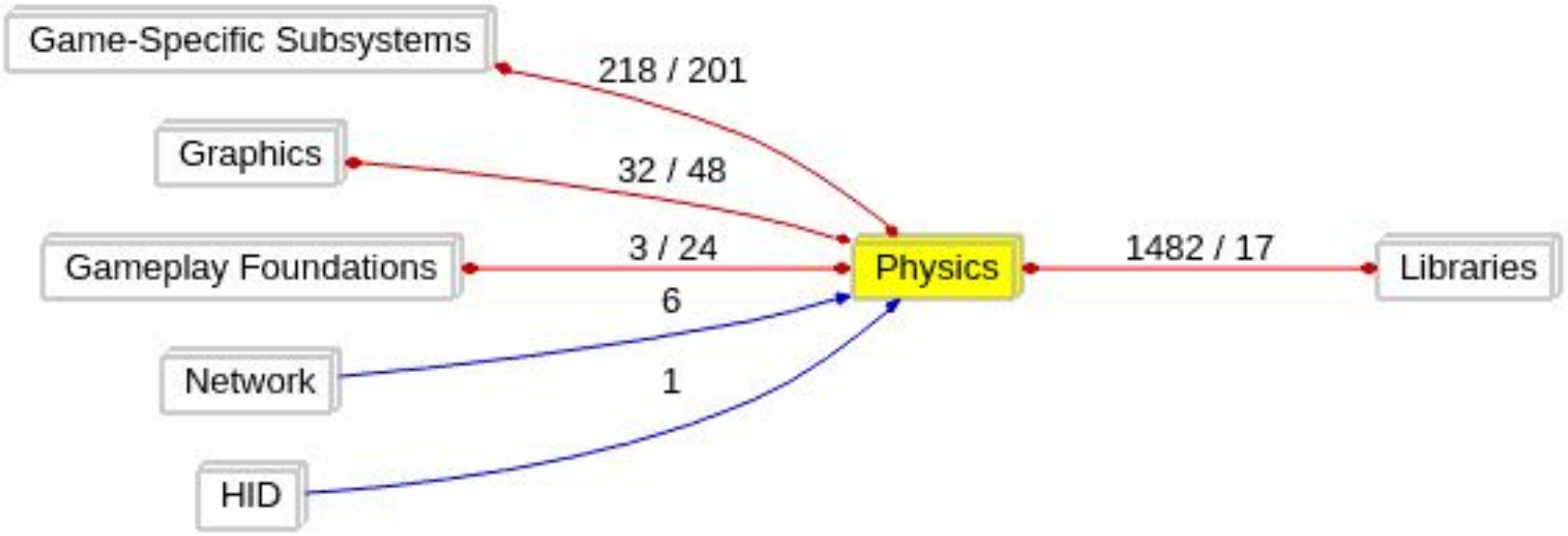


Dependencies: Network

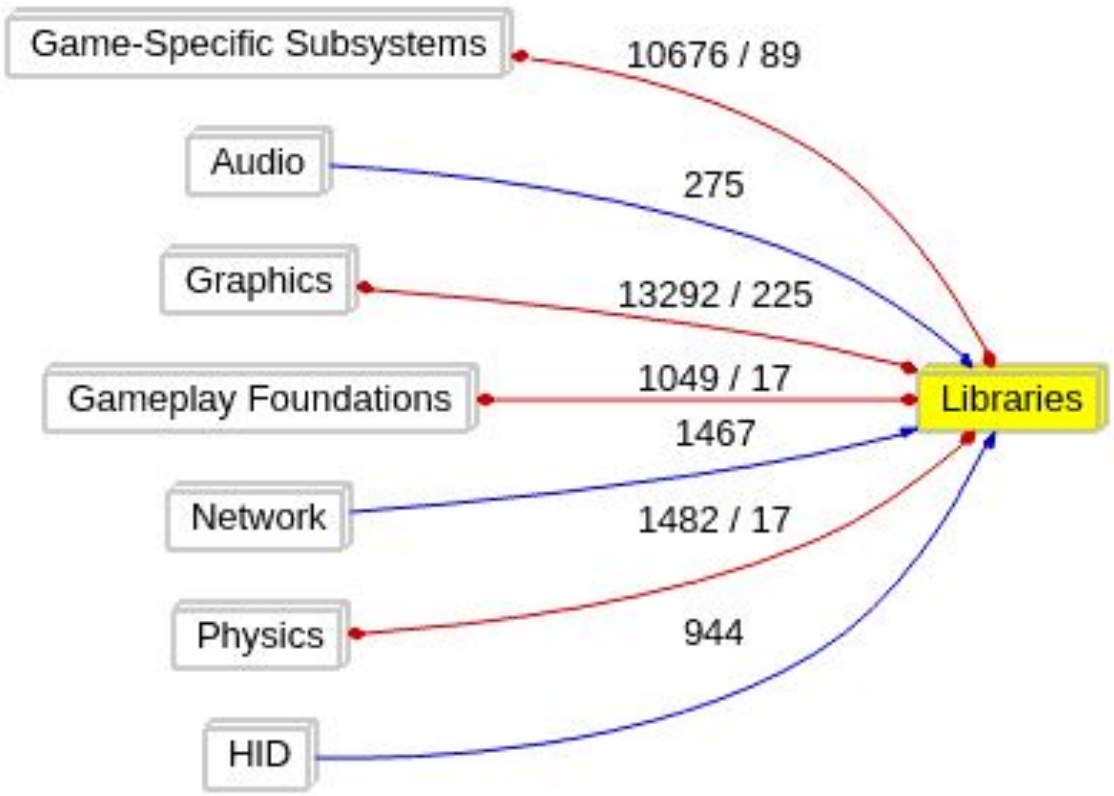


Dependencies: HID

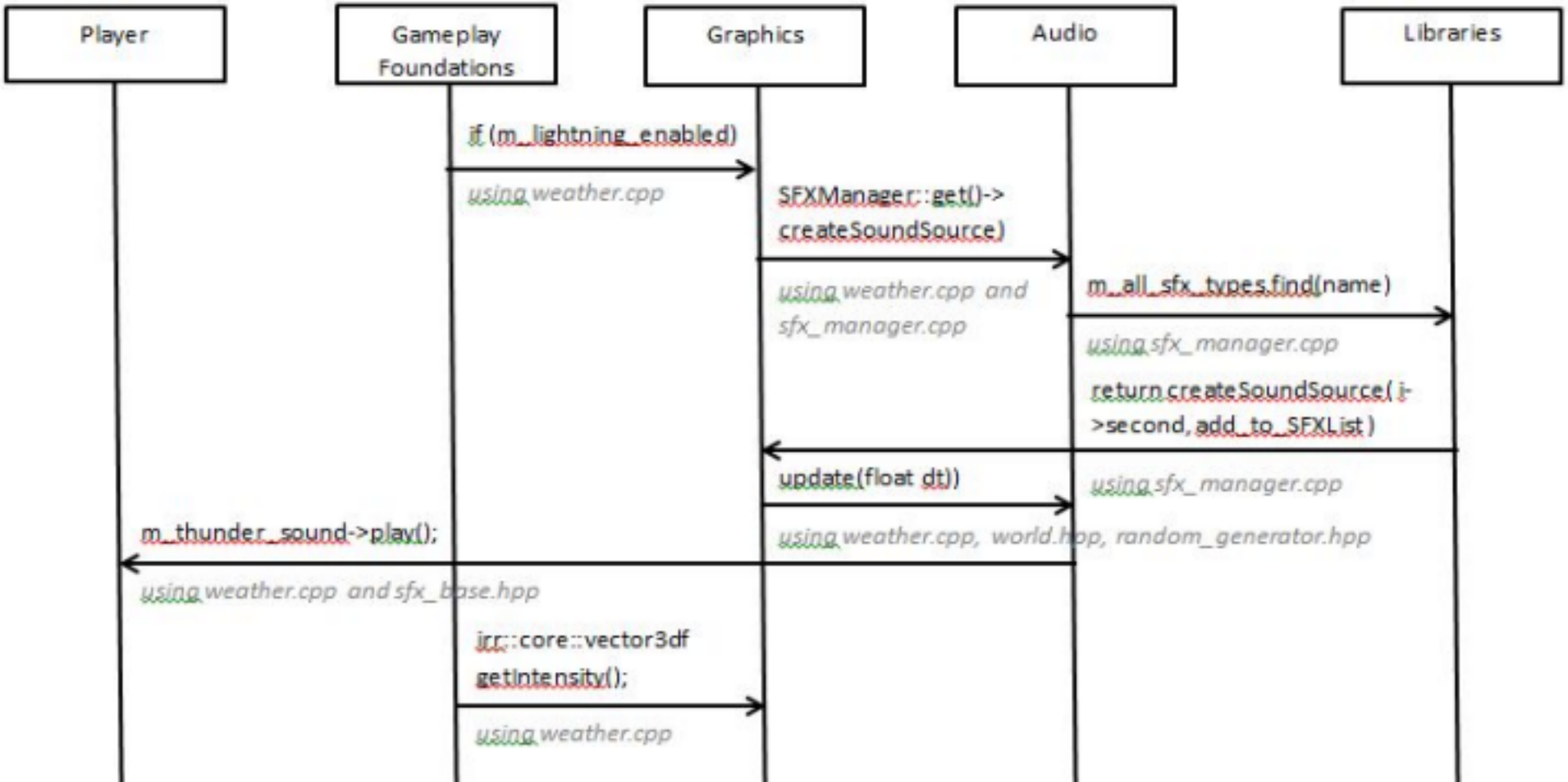
Dependencies: Physics



Dependencies: Libraries



Displaying Visuals and Sound of Thunder during a Race



Concurrency

- SuperTuxKart is designed to support multithreading and requires multi-core processors to be run optimally
- User input is an individual thread
- Graphics and Sound are also multi-threaded and are concurrent subsystems.

Limitations and difficulties

- General difficulty in distinguishing subsystems when inspecting the source code
- Usage of Understand and how the diagram is read
- While there are a fair amount of comments, some are nonsensical
- Many difficulties stem from the open source aspect of the game

Lessons Learned

- Obviously the developers didn’t plan out the architecture of STK, considering the amount of coupling (the Concrete Architecture diagram is very nearly a complete-graph of two-way arrows).
- Designing a Game Engine architecture is essential to mapping out possible issues in the future.

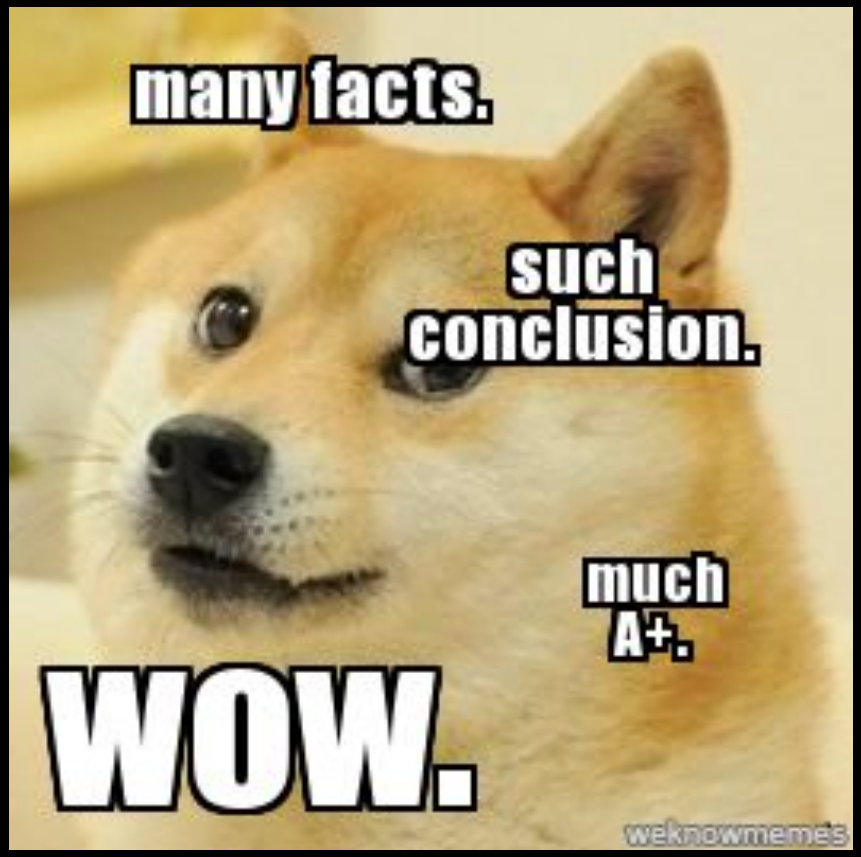
Architectural Style

We originally thought that STK would have a Layered architectural style, however, after looking over the source code, and seeing the amount of coupling, it is obvious that STK has an object-oriented style (with an extreme amount of coupling).

Conclusion

So to conclude:

- STK is an object oriented highly coupled architecture
- The source code has many apparent issues that derive from the fact that it is open source
- The conceptual and the concrete can vastly differ and make it quite difficult for someone just hopping into a project to know what's happening
- Understanding is a valuable tool that makes the entire process much easier



SOURCES

Gregory, J. (2009). *Game engine architecture*. Wellesley, Mass.: A K Peters.
Super Tux Kart Documentation: http://supertuxkart.sourceforge.net/doxygen/?title=doxygen
Super Tux Kart Source: https://github.com/supertuxkart/stk-code
Super Tux Kart FAQ: https://supertuxkart.net/FAQ
SciTools Understand: https://scitools.com/