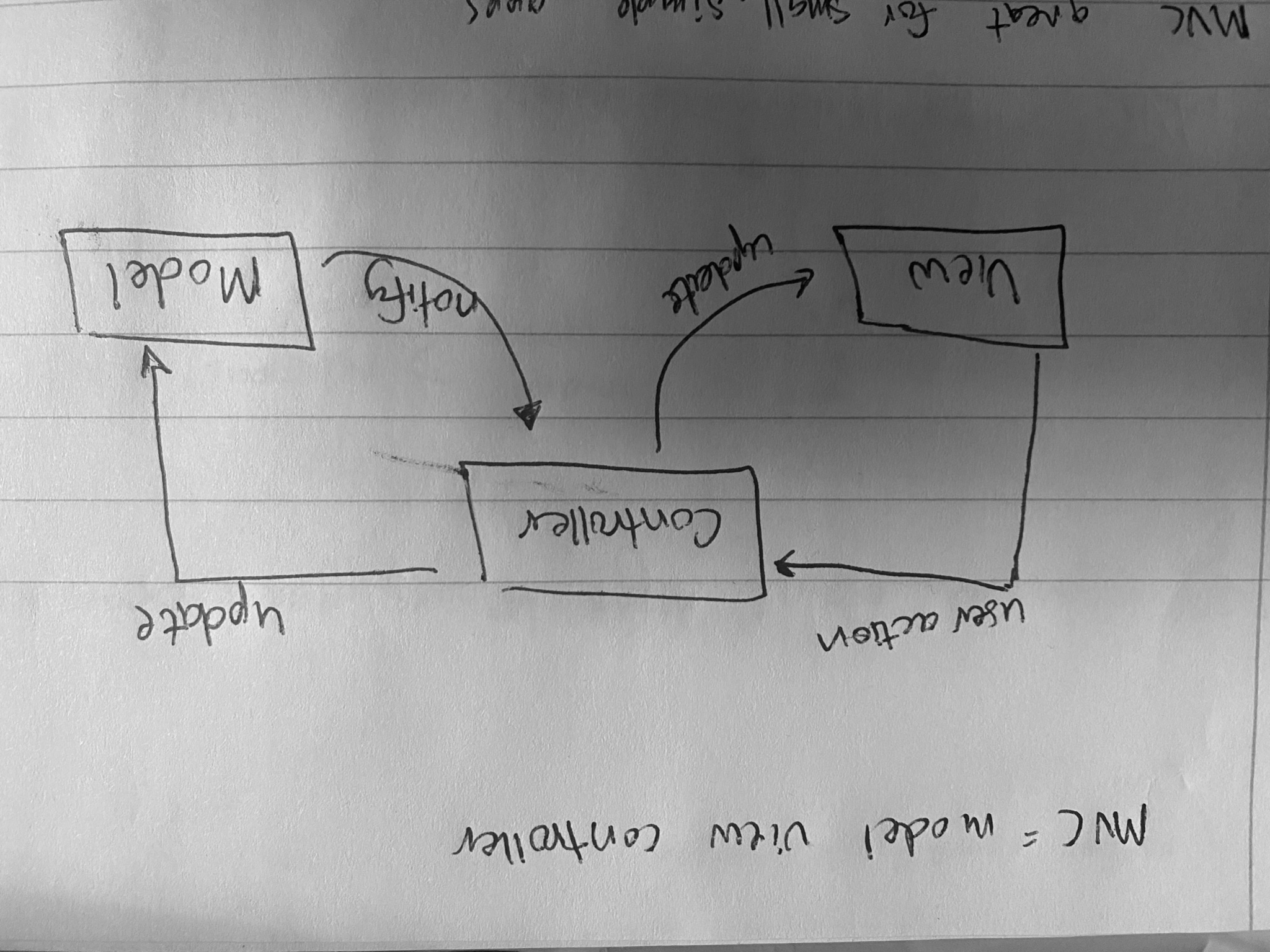
Apple’s MVC Pattern



Apple’s Model-View-Controller pattern is a means of distributing different workloads between the different sections. Each section has its own job. Implementing MVC pattern is a way of organizing and categorizing code in apps.This process works well for small simple apps, but not so well with more complex architectures when the lines of jobs become less concrete.

The view category, which is what the user sees, is in charge of the user’s interactions with the app. After the view category is the main portion, which is in charge of handling the apps data, network code, parsing, data sources, and delegates. Lastly, the controller is the middle man handling all the interactions between the two. It handles the life cycle of other objects within the application.

Some pros to using an MVC model is that code in the application is reusable, and interfaces are coded more efficiently. Because the application is smaller, the amount of code used to build it will be smaller; resulting in faster processing times. With larger applications where the amount of code, and jobs between the three are intermingled, processing speeds won’t be nearly as good.

After reading up on the different jobs that each category has, I think in small applications the MVC architecture works well. As applications become bigger and more complex the jobs between the three seem to start to merge, creating potential problems and bugs in code.

Sources Used:

1. https://developer.apple.com/library/archive/documentation/General/Conceptual/CocoaEncyclopedia/Model-View-Controller/Model-View-Controller.html
2. https://www.raywenderlich.com/1000705-model-view-controller-mvc-in-ios-a-modern-approach
3. https://shreysharma.com/benefits-and-drawbacks-of-mvc-architecture/