Durante el desarrollo del proyecto final de APO II, fueron varios los aprendizajes y los problemas que se lograron solucionar. Primero que todo, si bien trabajar en un grupo más grande (3 integrantes) a primera impresión parece facilitar todo, también tiene sus inconvenientes. Por ejemplo, la coordinación de tareas se complica un poco más, al igual que la comunicación. Es necesario, siempre que se pueda, reunirse personalmente los 3 integrantes del grupo (la comunicación y la toma de decisiones se hace más fácil en persona que virtualmente) para decidir cómo se dividirá el trabajo entre los 3 y para qué día se deben tener implementadas las funcionalidades establecidas.

En cuanto al desarrollo del programa, se evidenció la importancia y utilidad del desacoplamiento, principalmente durante la elaboración del multijugador. En una primera instancia, se buscó implementar las dos funcionalidades en una misma clase y dibujar la partida en la misma clase del Panel. Dicha decisión fue un error garrafal debido a que el sistema debía de hacer un número considerable de verificaciones tanto en el mundo como en la interfaz y los hilos, lo que producía problemas en la coordinación de los hilos generando múltiples excepciones (Debido a que hacia llamada a métodos que aún no habían terminado de ejecutarse. La solución fue hacer un desacoplamiento de los dos tipos de partida con la ayuda de herencias tanto en el mundo (Clase abstracta Partida de la que heredan PartidaIndividual y PartidaMultijugador) como de los paneles de la interfaz (Clase abstracta PanelJugar de la que heredan PanelIndividual y PanelMultijugador). En los hilos, también se manejaron clases diferentes para mover la nave de abajo, el disparo de abajo, la nave de arriba y el disparo de arriba. La mejora en rendimiento fue notable.

Por último, se presentaron problemas al hacer las pruebas de los archivos de persistencia, debido a que al probar dichos métodos y los que lo llamaban sobrescribía los archivos “definitivos”. La solución fue usar otra herramienta vista en clase: la sobrecarga. Dichos métodos anteriormente mencionados se sobrecargaron haciendo que reciban un String por parámetro, lo que generaba que los archivos de prueba se cargaran y guardaran en una ubicación diferente a los “definitivos” para que no los sobreescriba.