Design Paradigms

The design paradigm used by Team 7 for Project 1 was a combination of top-down functional decomposition, component-level design, object-oriented design.

Design paradigms aren’t singly utilized. Projects will almost always draw on different paradigms at the same time. For example, top-down functional decomposition, almost by definition, relies on component-level design. Top-down requires the system be broken into components. Component level design then specifies that those components should be modular, reusable, and contained. Zooming in further, object-oriented design is a paradigm about the organization of components.

One of the limitations of object oriented design is that instances of variables are contained within the object. We had to resolve this on several occasions in order to pass variables between objects within the program. For example, the ship count variable is selected in the UI, and the UI is controlled by the gamestate object. However, in order to utilize the ship count properly, the ship count needed to be passed to the player controller object so that the program could allow them to place the proper number of ships.

For many of the reasons listed above, we think Team 7 utilized object-oriented design, along with the other paradigms that accompany it. Many objects were used throughout the project. The highest level class, gamestate, held the global variables that controlled the state of the game. Beyond that, there were player controller objects, ship mesh objects, block meshes, level objects, and many more. Considering that Unreal Engine’s blueprints system is based of C++, it makes sense that this project would be object-oriented.