SE701: Assignment 3 | Design Changeability

Shane du Plessis
Department of Electrical, Computer and Software Engineering
The University of Auckland
sdup751@aucklanduni.ac.nz
10 May 2020

The design decisions will allow to support the following change cases relevant to this implementation.

Most likely:

- Initial seed count per hollow
- Hollows per player

Probable:

- GUI
- Rule modification
- Adding support for AI player

1. INTERFACES

Implementing an interface pattern is an advantage, as it means you do not have to alter any existing classes when adding new functionality (probable changes listed above). This chosen architecture adds a lot of flexibility to the design. Having the option to add or extend entire modules, makes it ideal for expandability and changeability.

2. ABSTRACT CLASS

To avoid cloned code, an abstract class was implemented, StorageElement will model the hollows and banks behaviour in the game of Kalah. Meaning, that if a change of behaviour needs to be implemented, only one of the now three classes will need to be changed (not two of two). Therefore, increasing its measure of changeability and reducing the chance of introducing error by not updating a duplicate piece of code.

3. STRUCTURE

Each class models an element of Kalah, making the implementation as modular as possible. By passing minimal parameters between objects, it promotes a flexible and scalable implementation. Less classes will be affected when introducing change, therefore, increasing the changeability of the overall design (i.e. no large classes responsible for more than one task).