In this research, we use reverse engineering through 4 + 1 views to understand and then remodel a legacy game application to MVC architecture.

The development of software includes many different developers focused on different tasks of the application. MVC architecture allows the different tasks to be isolated so that it is unnecessary for the developer of task A to fully understand task B within the same software system. This allows time to be saved in comprehension as well as changes in task A do not necessarily require changes in task B. In addition, the modularized code may be reused in different scenarios which would otherwise require rewriting the software.

A legacy game application of Rock Paper Scissors is reverse engineered with Enterprise Architect using 4 + 1 views with UML models. The different functionalities of the system are modularized into classes and methods based upon MVC architecture. The target system's code is then written using the explanation of a certain aspect of the software system, catering to the needs of different stakeholders in the integral software development process. 4 + 1 view can be implemented with an assortment of diagrams at each view with varying degrees of abstraction depending on the need. The purpose of 4 + 1 view in this research was to re engineer a legacy system to MVC architecture. Therefore certain UML diagrams at certain levels of abstraction were most beneficial, which are depicted below.

Reengineering

Software reengineering involves the renovation of existing software into a new form. Reengineering the selected legacy system consists of modularization and separation of tasks. Below is the process view (sequence diagram in UML) of both legacy and target systems showing how the functionality was separated into different classes.

**4 + 1 Views**

4 + 1 view is the framework around which the software system is modeled. Each view focuses on the explanation of a certain aspect of the software system, catering to the needs of different stakeholders in the integral software development process. 4 + 1 view can be implemented with an assortment of diagrams at each view with varying degrees of abstraction depending on the need. The purpose of 4 + 1 view in this research was to re engineer a legacy system to MVC architecture. Therefore certain UML diagrams at certain levels of abstraction were most beneficial, which are depicted below.

**Summary**

In this research, we use reverse engineering through 4 + 1 views to understand and then remodel a legacy game application to MVC architecture.

**Motivation**

The process of reverse engineering consists of deciphering software system design from a finished product. This process began with analyzing the use case diagrams for the legacy systems in order to construct the other UML diagrams in 4 + 1 view

**Approach**

A legacy game application of Rock Paper Scissors is reverse engineered with Enterprise Architect using 4 + 1 views with UML models. The different functionalities of the system are modularized into classes and methods based upon MVC architecture. The target system's code is then written using the explanation of a certain aspect of the software system, catering to the needs of different stakeholders in the integral software development process. 4 + 1 view can be implemented with an assortment of diagrams at each view with varying degrees of abstraction depending on the need. The purpose of 4 + 1 view in this research was to re engineer a legacy system to MVC architecture. Therefore certain UML diagrams at certain levels of abstraction were most beneficial, which are depicted below.

**Reverse Engineering**

The process of reverse engineering consists of deciphering software system design from a finished product. This process began with analyzing the use case diagrams for the legacy systems in order to construct the other UML diagrams in 4 + 1 view

**Flow of Entire Process**

An overview of the transition of different processes in order to arrive at the target system from the legacy system is illustrated below.

**Conclusion**

Reverse engineering using 4 + 1 views with UML diagrams provides an abstraction that allows for quicker comprehension of the legacy system. Re engineering code into MVC using this level of abstraction helps to prevent confusion of all the different interrelated parts. MVC architecture itself provides an intuitive organizational scheme that helps to structure the code in a meaningful way

**Reference**


