In a software development lifecycle in mobile application development, the maintenance stage is a stage that requires very large costs and time. This is indicated by the number of applications that must be removed by Google in their application store because they are not properly maintained. This shows that there are still many applications that are not prepared by considering the maintainability aspect. Previous research has shown that maintainability in object-oriented systems such as mobile applications can be improved by using appropriate design patterns. Unfortunately, this research does not describe in detail the analysis of the selection of each design pattern used and the effect of the selection on maintainability aspects. This study focuses on selecting a design pattern for a mobile application and looking at its effect on maintainability such as analyzability, changeability, stability, and testability. The application source code is analyzed to determine the existing design problems and their effect on OO metrics, then look for the appropriate design pattern to see the effect of using several design patterns separately, as well as the combination of these design patterns on maintainability aspects. From the analysis found 2 things. First, the selection of design patterns in mobile applications, especially Android, can be done by looking for design problems with adjustments such as changing the constructor into methods that support the activity lifecycle, and not all design problems can be solved, such as the number of parent classes from the Activity which basically exceeds the accepted limit. The two implementations of the design pattern can improve the maintainability value of the selected mobile application between 11.35% to 17.83% because it can apply separation of concern. However, the stability aspect can deteriorate because it is necessary to add classes to increase dependencies.

Keywords: design pattern, mobile application development, maintainability, non-functional requirement, software engineering