One of the GRASP heuristics used in developing StockMATE was the principle of Low Coupling. Coupling is the concept of how strongly an element depends on other elements of the design. Coupling also infers the interconnectedness of objects and even awareness of them. A software package that has many highly coupled, intertwined objects would be very difficult to change; since small adjustments could ripple through the entire design and could also lead to unforeseen bugs and broken functionality. The ideal is design with Low Coupling Pattern, so elements can freely act independent of one another. This also affords allows better understanding of the intended purpose of the elements, as well as gives option for reusability across the same or perhaps entirely different projects.

StockMATE strives to adapt the Low Coupling pattern across its design. For instance, the NestedMap Class has zero concept of almost every other class in the design, save FilingMap. FilingMap implements the NestedMap for the ease of storing and retrieving filing data, but no other classes know what NestedMap is. NestedMap could be altered, perhaps at a future date it is decided that a different approach for storing the Filing data is preferred. The only other class that would have to change would be FilingMap. NestedMap could also be brought into another, completely different project and reused in its entirety.

Not all elements in the StockMATE design have this single relationship with another class. Larman recognizes as an element as having “too many” relationships context dependent. As an example, the the class with the highest amount of the relationships in StockMATE is the Controller class. However this is intentional by design, actually following another GRASP concept: Controller. Where this class receives and coordinates the system operation of the major subsystems. By pure virtue of this pattern, this class must reach out and connect the major players and pass messages to accomplish the desired use cases. The Controller class in StockMATE is the root object, the conductor of all the moving parts between the abstracted methods and passing data.

An overemphasis of low coupling can lead to small number of classes that do too much. This steps on the toes of another GRASP heuristic: high cohesion. High cohesion is the patter to focus object design into manageable and understandable roles. Recognizing this with StockMATE some late development adjustments were made to tone down the activities of relatively large class: FilingSummary. Early design concepts had this class owning the responsibility of assigning filing and decoding filing tags. This was done in efforts to apply the low coupling effort. However, a balancing act must be achieved between low coupling and high cohesion. By requiring FilingSummary the responsibility of managing filing tags in addition to summarizing the data of the filings was simply too much, so it was decided to separate the two responsibilities into two different classes. Thus, achieving the goal of low coupling while maintaining high cohesion.

Good software design needs to have a global conscience of all GRASP concepts in implementation. Some of these patterns were discussed here however the focus for StockMATE’s was low coupling, high cohesion and controller implementation.