1. API modification made prior to the last deliverable (unit test suite), but was not documented at the time:

We removed the checkValidLocation() method of class ChessBoard because the validity of a location can be verified by itself, and it has nothing to do with the chessboard. We check the validity in the constructor of Location. If the parameters are not valid, an exception is thrown. In practice, it is impossible for a location to be illegal during the normal execution of our code, and this only acts as a preventive measure in case the software is to be modified in the future.

1. Additional tests for internal methods:

We add unit tests for outrank() in the class Piece. We also make clear() in ChessBoard private and modify its tests. In our last submission, we made clear() public only because we need to access it from unit tests, and we did not test outrank() directly because it is protected.

In order to obey the minimum visibility principle and provide a complete test suit at the same time, we now revise our design and use reflection to test methods that are not public.

1. How we could have avoided the revision

First, there had inconsistency between the API document and our actual code, mainly because we failed to update documentation in time. To avoid this, changes should be immediately added to the documentation after they are made.

Second, our unit tests were not complete because we did not know how to test private and protected methods. We also made a method public for testing purpose, which is not good for the encapsulation. To avoid this problem, we should use reflection to test non-public methods in depth, and keep the access rights minimum regardless of the test component.