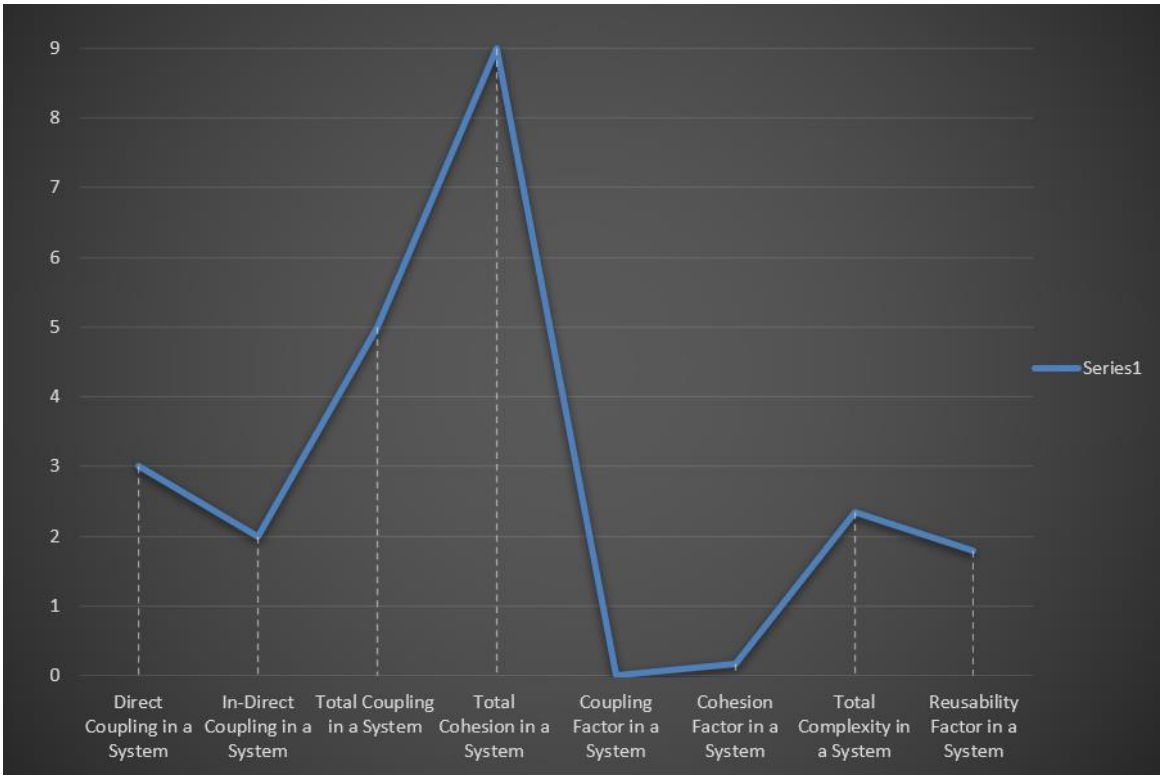
Software Engineering

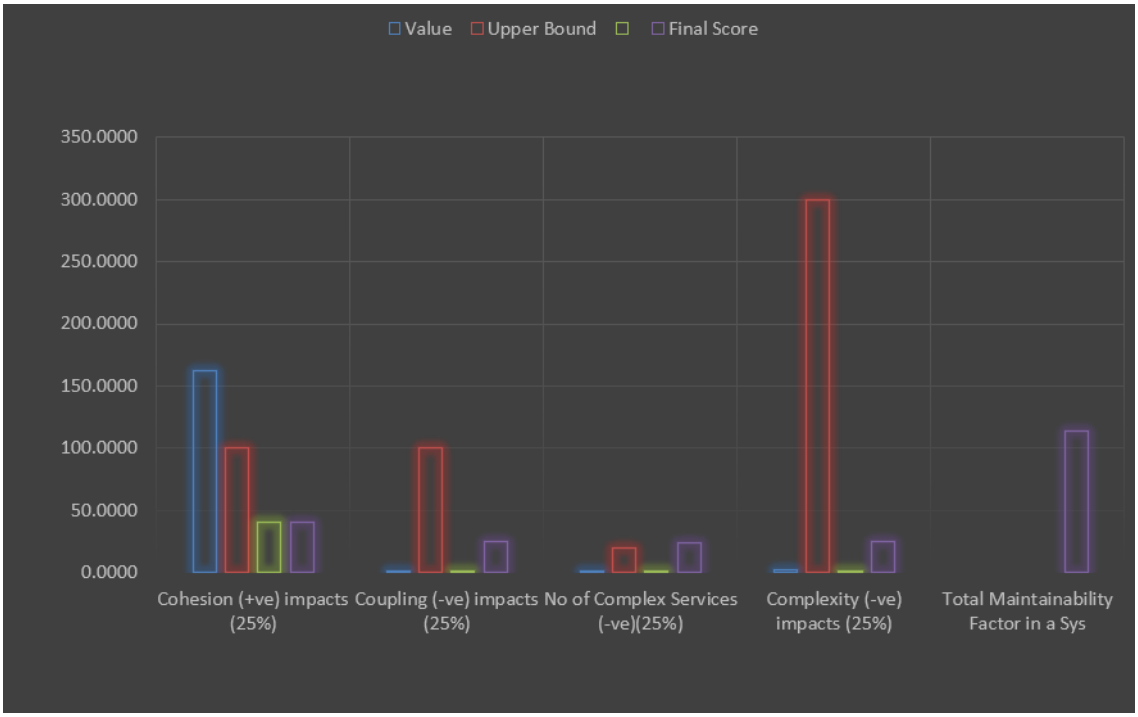
Assignment 7

Asad Zaman (18p-0034)

Sana Haider (18p-0011)

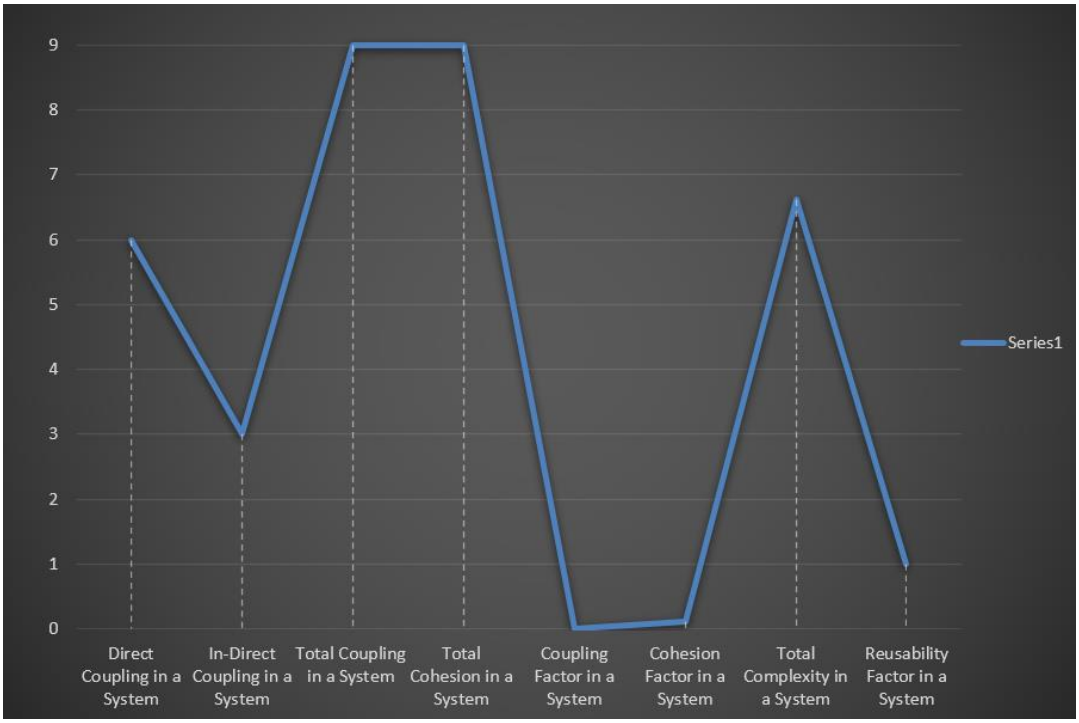
1. Architectural Evaluation Using Metrics:

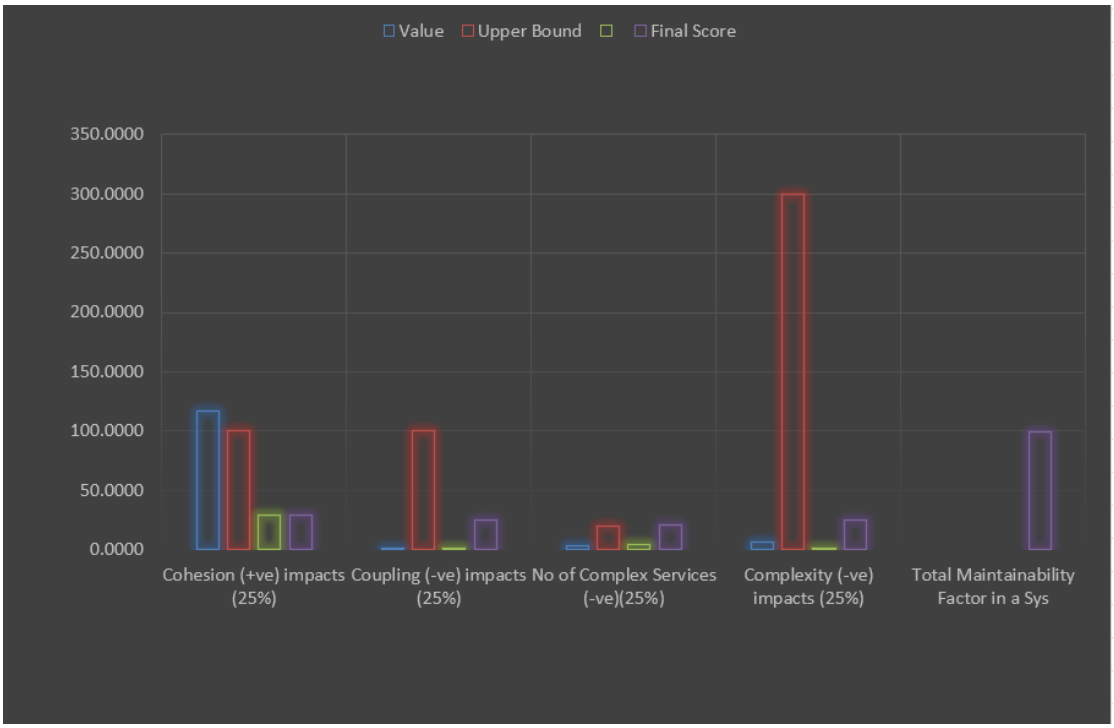




There is very little coupling and high cohesion in our software architecture, which is a positive thing because this will increase the maintainability factor, meaning that the software will be easy to maintain. The value of complexity is low than the upper bound. The little the complexity the more reusable and maintainable the software is.

2.





Our whole goal was to implement architecture that would be maintainable even if we add new services, when we added new services with more functionalities the effect on low coupling was very little that is, it almost remained the same the cohesion decreased a little but its still greater than the better which is a positive thing for our software, we made sure that none of the SOLID principle are being violated while designing the software. So even if we add new services and functionality our software will have good performance just like we wanted it to have.

Complexity:

Our proposed Architecture has a very little complexity than the upper bound. This is due to the fact that we will be focusing on code and algorithms with low time and space complexity, so it is efficient enough to have low complexity. The coupling in this architecture is also low which will lead to low complexity, the services will be independent from each other and hence reduce the complexity.

Reusability:

The reusability factor decreased a bit when we added a new service with many functionalities, but it's still reusable when we have services-oriented architecture because all the services are independent from each other to most extends, so we can reuse the code to achieve reusability. Low coupling and high cohesion will automatically lead to reusability, so in future if we want maintain some services or if we want to add

new functionalities to the software it will be easier to understand the code and to make changes to the existing code since it won't effect other services and functionalities.

Maintainability:

Providing the budget we could use Continuous Integration to support maintainability of software in terms of different builds, testing etc. We could also use some principles of DevOps. Maintenance would be crucial to our software. Maintenance in terms of management would be also necessary, related to customer feedback, new customer needs, bugs in our software while in operation. And we would also need to deal with technical maintenance like fixing bugs in time without affecting operation time, testing new added services and existing services each time our software is in operation or a new build is produced. We would also need to provide regular patches to our software so our software works effectively. We would also need to be wary of cyber-attacks as security is very important so our customers' credentials do not get stolen. So, for that we would also need to maintain our software's security.