**4.**

**Tactics for Usability:-**

In our system, tactics for usability is implemented. This tactics ensure to implement quality attributes like mobility, allocation of system responsibility, scalability.

The software system is independent of the platform like android, IOS and windows which incorporate mobility of the software and enhances the usability.

Allocation of the responsibility ensures that additional system responsibilities have been allocated, as needed, to assist the user. The system is designed in a very user friendly way so that it is very easy for the user to learn how to use the system and how to efficiently achieve the task at hand and how to adapt and configure the system.

The evaluation engine which consists of information extractor, context store & parse, query parser etc is always being improved by a new input/feedback provided by the user which is improving the logical unit of the entire system. By this means the horizontal scalability is achieved and the system is able to reply the user in more contextual way.

**Tactics for Modifiability:-**

Tactics to control modifiability have as their goal controlling the complexity of making changes, as well as the time and cost to make changes. In our system, there will be always room for enhancing the model of contextual reply. Every time the feedback taken from the users are stored in adoption model which will help to improve the conceptuality of the learning algorithm in turns it will help to enhance the context store and parse unit of the query parser.

**Tactics for Performance:-**

The goal of performance tactics is to generate a response to an event arriving at the system within some time-based constraint. Performance tactics control the time within which a response is generated. The speech parser in our system is efficient enough to understand the user’s input irrespective of the accent and parse the phonemes in parallel which helps to achieve the multitasking usability. The learning algorithm of the evaluation model returns the contextual reply very fast. In this way the system is not only capable of handling multiple queries by multiple users at a time but also can response each user in very small time like 2 seconds.

**2nd part:-**

Tactics for modifiability will be preferred over tactics for performance and usability. The reason is as the system itself is flexible enough to be modified by learning new things with almost no extra time and no excessive cost. It in terns helps to improve the performance of the system as well as the quality of the performance (i.e. The system is able to reply more contextually).

**5.**

If we think about how to implement tactics to provide security in a system, we can have four categories of tactics: detect, resist, react, and recover. Adversarial Attack, man in the middle attack are very common cyber security attack in Machine Learning based systems [including reinforcement-based ones](https://arxiv.org/abs/1701.04143).

From a defender perspective, this type of attack has proven (so far) to be very problematic because we don’t yet have an effective way of defending against such attacks. Fundamentally, we don’t have an efficient way to get DNNs to generate good output for *all inputs*. Getting them to do so is incredibly hard because DNNs perform nonlinear optimizations within very large spaces and we have yet to teach them to learn high level representation that generalize well.

Due to the difficulty, detection-based defenses have attracted a lot of attention recently as alternative solutions.

A class is introduced by Scientist Grosse is solely for adversarial examples and an additional binary classifier trained by scientist Gong to decide whether an instance is adversarial or not which will be implemented with the learning algorithm in our system to detect such attacks.