Architectural Thinking for Intelligent Systems Assignment 8

Team 11

1. Which 2 architectural principles are the most important ones when developing the system architecture?

   For our system we have selected the following 2 important architectural principles:

   - Modularity
   - Incrementality

2. Why did you decide for these principles? How do the two selected principles help in achieving the desired system qualities?

   - As our system is very **large**, **coupled** & **highly complex** to implement, we go by our intuition to make the system modular and then implement these components using a **Parallel Development model** (Incremental). Due to the architecture we have envisioned for our system, we cannot use Loose Coupling or High Cohesion principles directly. The modules/components in our system are very tightly coupled i.e. changes in 'Speech/Text Recognition' module would severely affect functioning of 'Evaluation Engine' & 'Model Adaption' modules.

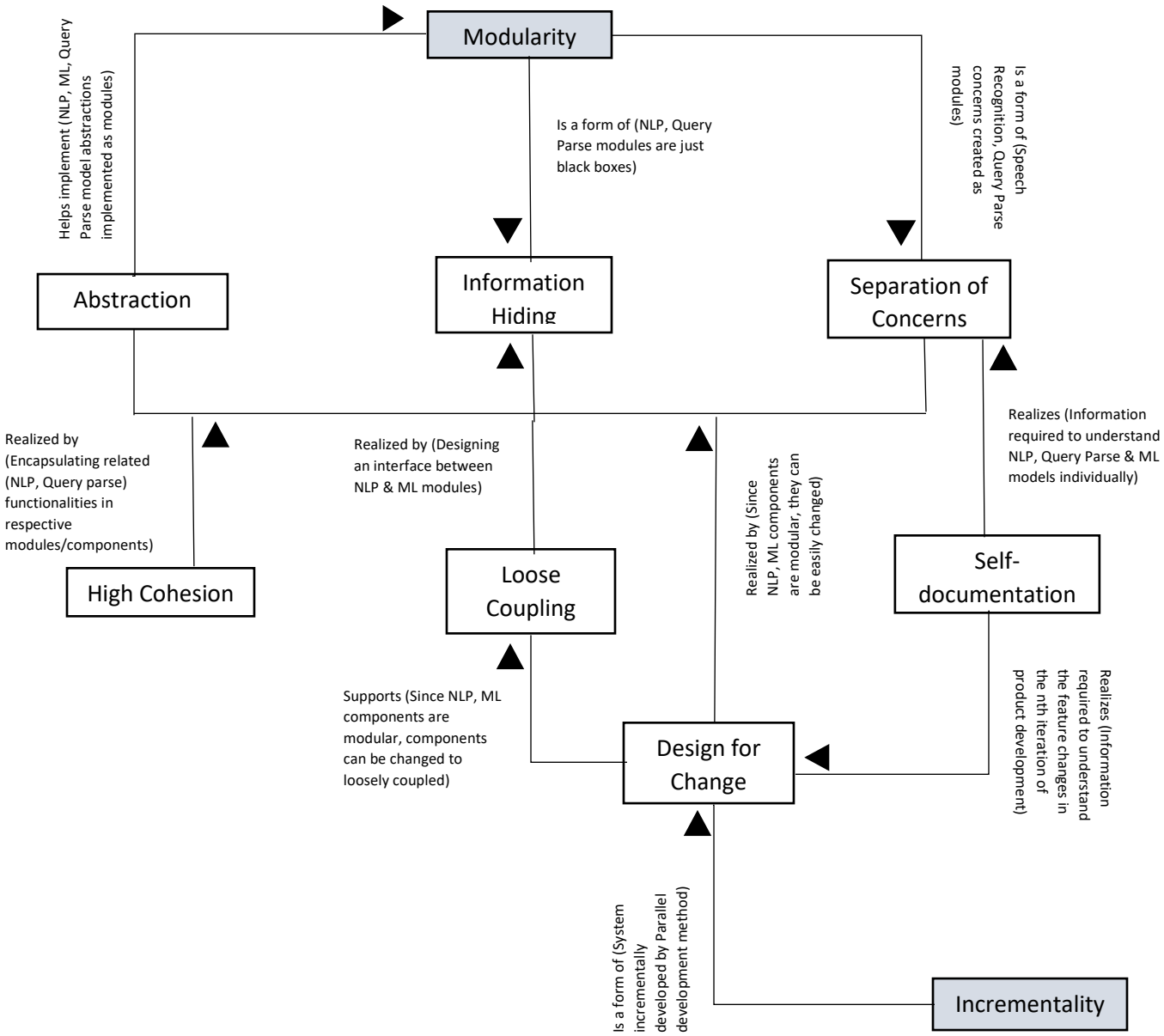
   - **Modularity:** In brief, as our system is multi-faceted & utilizes techniques from various disciplines such as Statistics, Machine Learning, Data Engineering & generic Software Engineering. We could modularize each functionality based on the technology used and then hire different specialists from the above mentioned fields and employ them to design & develop their respective components in parallel. Further, this modular approach would fasten up the development process, reduce complexity and increased understandability of the system and also paves a way for evolutionary improvement of the modules.
     Example: We can modularize the recommender system into Speech/Text Recognition, Evaluation Engine, Context Store & Model Adaption components respectively.

   - **Incrementality:** There is a strong dependency between all the modules in our system, as a result of which the development of all the agreed functionalities in one shot would be a nightmare (may lead to poorer accuracy/performance & in some cases an overall system failure too). In order to tackle this and improve the performance of the system systematically, we could use Incrementality principle to design & release the system with a set of functionalities, evaluate results & feedback from the stakeholders/end users during every iteration and then repeat the process again with newer features and performance improvements.

Example:

a. We could provide Multi-lingual support incrementally to avoid potential bugs/issues which would spoil the overall experience of the system altogether and etc.
b. The machine/deep learning model could be tweaked or changed altogether in the forthcoming iterations after analyzing the model accuracy and user feedback.

3. Draw a specific instantiation of the diagram of slide 19 of deck A9 to show how other principles can support those two principles, which you selected. Annotate the relationship arrows in the diagram with specific ideas/solutions that you will apply when implementing the principles in the architecture.



4. Select two tactics to implement one or several of the quality attributes. How does the tactic support the quality attribute? Compare the selected tactic to at least one other available tactic and explain why you preferred one tactic over the other.

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.

Mobility: The system is platform independent (Android, IOS) 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 accuracy for contextual reply. Every time the feedback collected from the user, enhances the knowledge generated in the context store which in turn helps to improve the model performance in in terms of question answering & recommendations to user query. To summarize, we can generalize/normalize the data, alter the learning rate, iterations and the model entirely and so on using this tactic.

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 should be efficient to understand the user's input irrespective of the accent and parse the phonemes in **parallel** which helps to achieve the multitasking usability & reduce false positives. Further, the accuracy of the system (major quality attribute) can be increased using various techniques such as Cross-folding, Normalization of data, Increased Compute Memory resources etc. Thus, 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.

Tactics for **modifiability** will be **preferred over** tactics for **performance** and **usability**. The reason is, the system should be flexible so that it can be modified as per requirements (such

as the learning model, query parser and so on) and to do this we would be exploring different tactics for modifiability. This in turn helps to improve the overall performance & accuracy of the system. In most cases, the higher the accuracy is, the more usable and preferred the system is among end users.

5. Cybersecurity is a very important quality attribute of every software. What tactic(s) do you use to protect your system and why?

   If we think about how to implement tactics to provide security in a system, we can have four categories of tactics:

   - **Detect**
   - **Resist**
   - **React**
   - **Recover**.

   Adversarial Attack, **Man in the Middle** attacks are very common cyber security threats in end user Machine Learning based systems. From a defender perspective, this type of threats 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 introduced by renowned scientist 'Dr. Grosse' is solely for adversarial examples and an additional binary classifier trained by scientist 'Dr. Gong' to decide whether an instance is adversarial or not will be implemented with the learning algorithm in our system to detect such instances. Further, as most of the computations are handled in the compute nodes in the cloud, we can use tactics to hash the query contents or encrypt the communication channels between the servers and the end user device to prevent common threats like Man in the middle attacks and so on.