Architecture chosen for the system and why?

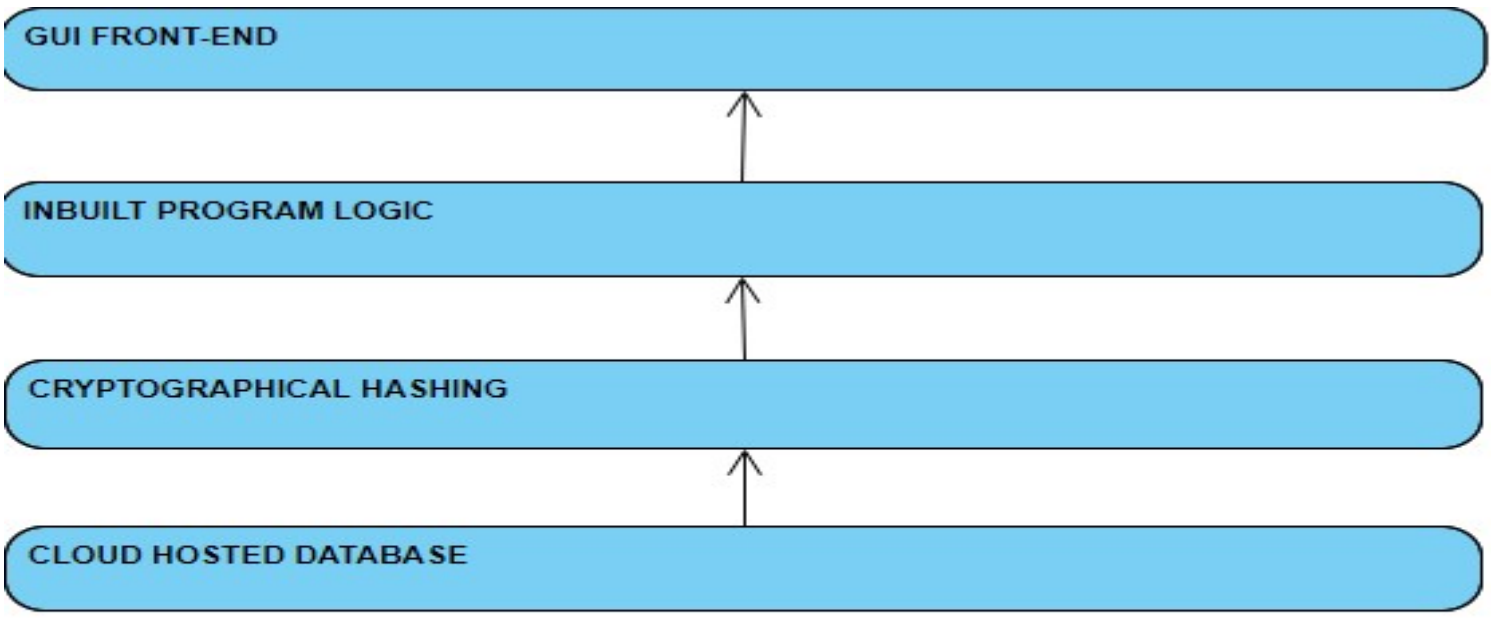
The software architecture which is going to best suited for supply chain management is going to be the layered pattern. It is one of the most well-known software architecture patterns.

In case of the layered pattern the main idea is to split the whole software into layers where each layer is going to contain a certain responsibility and provides a service to a higher layer.

The layered pattern is most suited for this SCM software because the software can be mostly broken up down into a number of parts: -

- GUI Frontend
- Cloud Hosted Database
- Inbuilt Program Logic
- Cryptographical Hashing

At the lowest level would be the cloud hosted database that is going to contain all the product details and is the main part from where all the details can be queried. Next is the cryptographical hashing codes for each and every product so that proper data integrity is followed and the various products are just referred by a vague code in order to protect the company's R&D products from competitors. Next would be the inbuilt program logic that is going to first of all going to interface the GUI Frontend with the cloud hosted Database and help query the details. Also, at the top would be the GUI Frontend in which the data can be filled and the details be queried back and shown.



LAYERED PATTERN DIAGRAM

Initial Architecture

Modifiability

The modifiability of a software system is the ease with which it can be modified to the change in the environment, requirement or functional specification.

In the 21st century due to increasing demands of regular customers, the SCM should be ready in order to be modifiable to serve the vast number of customers who as a whole depend upon the SCM in order to fulfil their requirements. To make the system modifiable, the factors need to be carefully studied and then the functionalities can be added on different layers, respecting the functional specifications of the system. Modifiability is one of the key features of our system.

To check if the modifiability is being correctly applied to our system, ALMA (Architecture-Level Modifiability Analysis Method) method is utilized to make our SCM as robust as possible.

ALMA has a structure consisting of the following five steps:

1. Set goal: determine the aim of the analysis

2. Describe software architecture: give a description of the relevant parts of the software architecture

3. Elicit scenarios: find the set of relevant scenarios

4. Evaluate scenarios: determine the effect of the scenarios

5. Interpret the results: draw conclusions from the analysis results

Tactics for Modifiability

1. Reduce Coupling
2. Increase Cohesion

Reducing Coupling

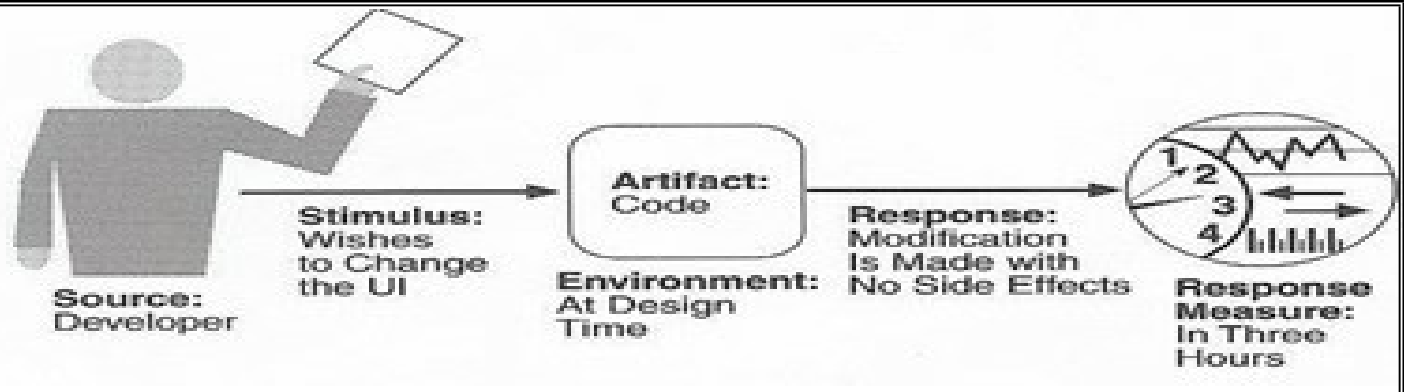
Modules in systems have different responsibilities. If the responsibilities of modules overlap, a single change request may impact multiple modules. The probability that a modification to one module will propagate to another is called coupling. Coupling is an "inter module" characteristic and is the enemy of modifiability: low coupling is good.

In our system if one component gets failed it should not affect the whole system and this is why we are using layered architecture. The layers which are critical would be placed at the last and also additional backups will be maintained in order for the system to fall back on in case of any failure. This would lead to a decrease in the coupling of the system.

Increasing Cohesion

Cohesion: Cohesion measures how much the responsibilities of a module are related. The cohesion of a module is the probability that a change request to one responsibility will impact another one.

Cohesion is an intra-module characteristic and high cohesion is good. As our system follows layered pattern so the related functions are grouped together in a single layer as a whole which increases the cohesion of the system.



This is a six-part diagram which shows that how the modification takes place when the change is made in the UI of any system. In respect to any SCM system the modifiability attribute can be worked upon as: -

Source: Developer

Stimulus: Wish to change the database from local datacentre to cloud for access by all the stakeholders

Environment: At Design time

Modification: Modification is made with No side effects and provides extra accessibility to all the stakeholders

Response Measure: Time depends upon the amount of data that needs to be migrated.

Artifact: Code and Cloud

Availability

Availability refers to a property of software that it is there and ready to carry out its task when you need it to be. This is a broad perspective and encompasses what is normally called reliability (although it may encompass additional considerations such as downtime due to periodic maintenance).

Our system is based on a real-time use-case, so availability is the foundation and the key factor for the software as it needs to be available for all the stakeholders in the system. With the increase in number of stakeholders the system needs to be available for serving the stakeholders who may use the SCM for any work they want to have with it. The features available in the software have to accommodate the various functionalities that are required for the system, so the availability is enhanced for the same.

Tactics for Availability

1. **Ping/echo :** One component issues a ping and expects to receive back an echo, within a predefined time, from the component under scrutiny.

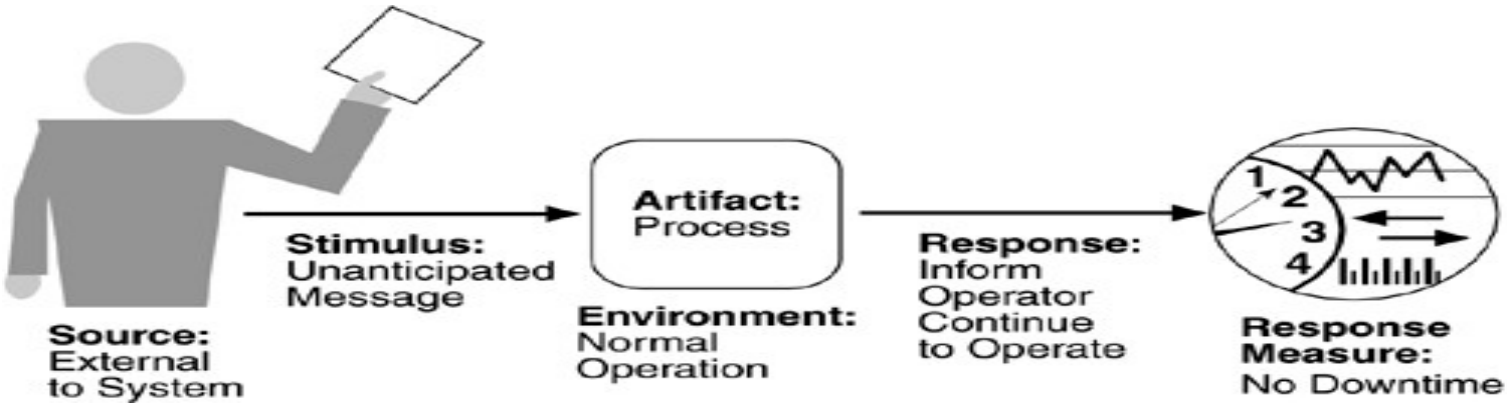
We use this to find the position of the particular raw product in the system with the help of proper secured technology and on a later stage we plan to add blockchain using which we would be able to know the current position of the product.

2. **Spare :** A standby spare computing platform is configured to replace many failed components. It must be rebooted to the appropriate software configuration and have its state initialized when a failure occurs.

In case if the system fails then this spare comes into use so that the services don't come down to stop the function of the system. To avoid the failure, we keep a spare system as a backup so that in case a fault occurs the spare system can easily take the faulty system's place and no issue gets created for the stakeholders.

3. **Checkpoint/rollback :** A checkpoint is a recording of a consistent state created either periodically or in response to specific events. If System fails, in this case, the system should be restored using a previous checkpoint of a consistent state and a log of the transactions that occurred since the snapshot was taken.

The software's database is going to be hosted on the cloud so that the stakeholders can easily access the software from around the globe but also the details regarding the software's performance and other system logs would be stored in a centralized datacentre locally so that in case of a severe failure a checkpoint is created for instant rollback. The checkpoint or rollback system helps to back date the system if any failure happens to it. It also keeps the track and record of the system transaction which avoid the data loss.



Source of stimulus : System

Stimulus : Fault with credentials of a user or the product id

Artifact : Processor, communication channel

Environment : System reset when the fault or failure occurs

Response : Notification passed to user to fill appropriate details

Response measure : Immediate repair to previous state

Performance

Indication of the responsiveness of a system to execute any action within a given time interval. Performance is concerned with how long it takes the system to respond when an event occurs.

In our system, tracking facility is provided to track the available raw materials which are available for manufacturing. So, performance of the search feature of the products would be greatly affecting the functioning of the application as there would be thousands of raw materials which would be available as a whole for the system. Also, the responsiveness in addressing the user searches plays a very vital role for the system. The search query should be passed on by quickly communicating and generating the results from the database. As per our system, the performance is enhanced for overall smooth functioning. Quick response time is set to handle all the query and results accurately.

Tactics for Performance

1. Resource Demand

- **Increase computational efficiency**.

  As per our system, we have improved the algorithms used in important application critical areas such as search facility, product tracking so that the software can easily serve this request and as a result it will decrease latency.

- **Reduce computational overhead**.

  If there is no request for a resource in the application, processing needs are reduced to improve the performance and also this leads to further reduction in loads.

- **Manage event rate**.

  If there is no control over the arrival of externally generated events, queries are sampled at a lower frequency and later fed into a queue. After that queries are then taken from the queue and as a whole the software can behave normally at even time of peak requirement.

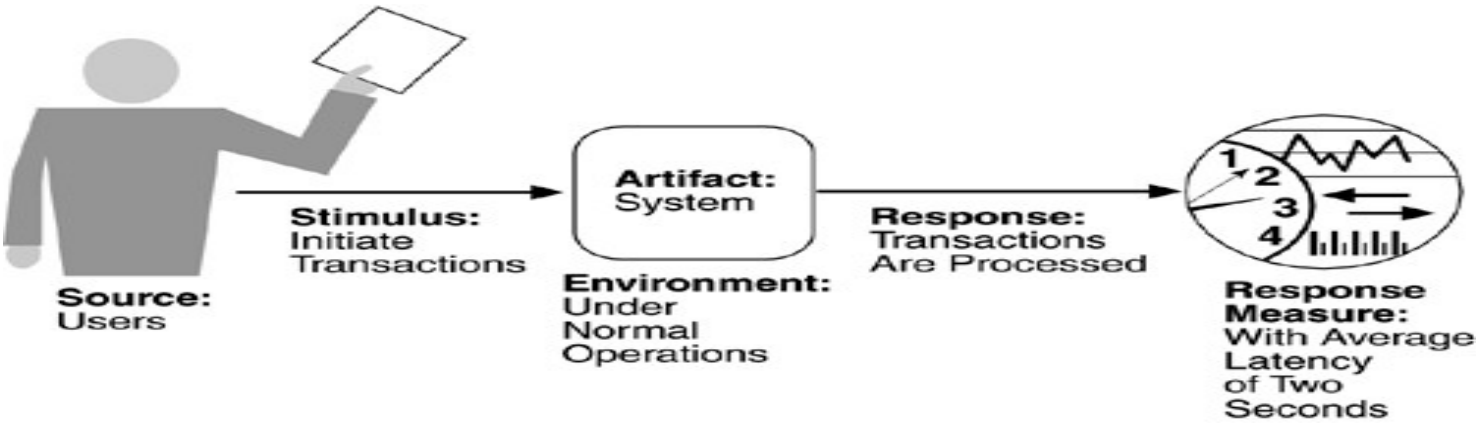
  The execution times and queue sizes are bounded by a certain limit so that no queries are handled in a haphazard manner during heavy traffic of queued queries. The resources are processed in such a way.

2. Resource Management

- **Introduce concurrency.**
  To reduce blocked time, we have configured the system in such a way that queries are processed parallelly. Concurrency is introduced in various blocks of the application and as a result improves the overall performance.
- **Increase available resources**.
  Faster processors, additional processors, additional memory, and faster networks are equipped in our software in order to improve the overall performance.

3. **Resource Arbitration**

- **Scheduling Policy**
  As per our application, we firmly believe that scheduling could be done on first-in/first-out manner. This would guarantee processing of queries in a fair manner. Also the scheduling would be done in such a way that the queries would be sorted one after the another as a whole so that it improves the overall performance of the system.



Source of stimulus: Internal sources.

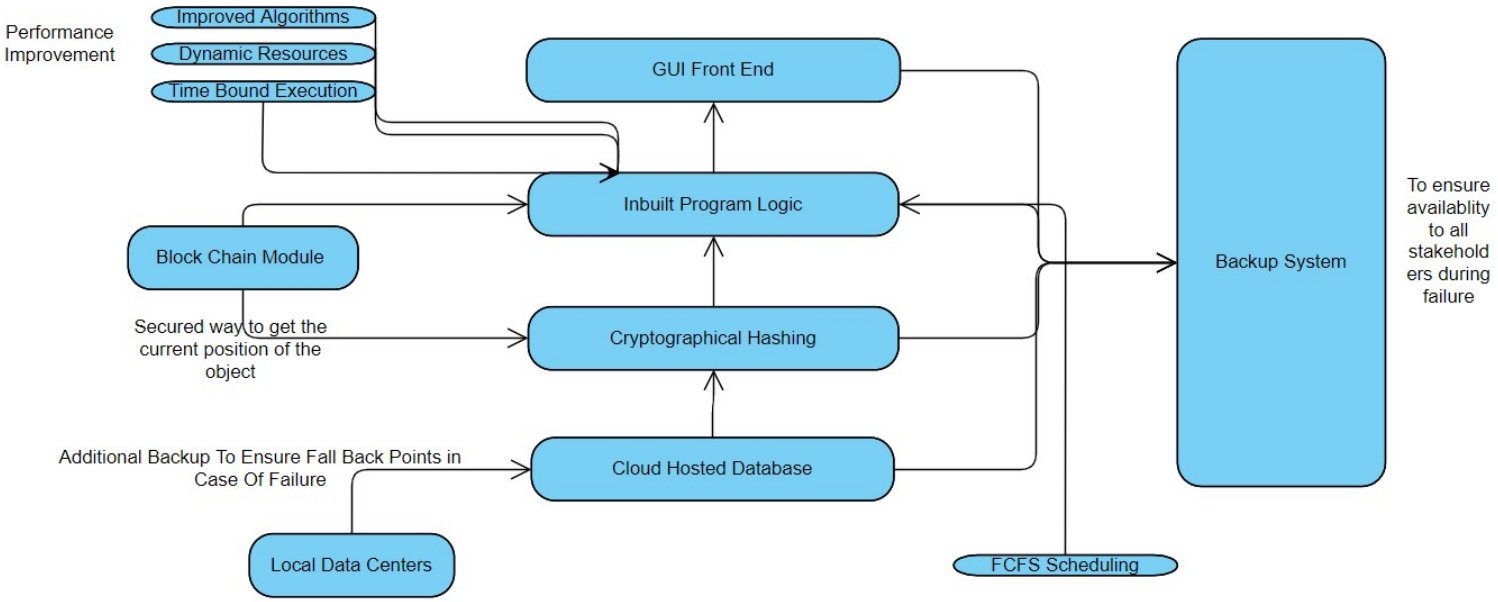
Stimulus: The stimuli are the arrival of the queries.

Artifact: System's services

Environment: Normal

Response: The queries are processed immediately

Response measure: Two seconds but can depend on complexity



Refined Architecture of the system