**1. Project Overview**

Message processor application which receives the sale notification messages and maintain an inventory of the sales.

**1.1 Objective**

Message processing application needs to be created with the set of requirements (mentioned in the requirement section) and should have the capability to provide inventory information’s like product type, total sales count and sales value of the product by extracting the meaningful information’s from the plain text message received.

**1.2 Business Case**

This application has the capability to read regex pattern based (pattern must be defined) sale notification messages and extracts the business information’s like product type, sale value and sale count, adjustment operations if any etc.

**1.3 Risks**

Any message received other than the defined regex patterns may create unexpected behavior of the system.

**1.4 Out of Scope**

Currently this application reads the inputs from the List of Strings only, but still any enhancements like reading a message from excel/csv/plain text files or any Rabbitmq queue calls from the other systems can be done by simple utilities.

**1.5 Limitations:**

Due to the external jar usage constraint, I could not use log4j to log info/warn/error messages, as I have used apache commons for lot of utilities and Junit jar for junit test cases.

**2. Technical Specifications**

Any Java 8 installed machine with maven would be enough to run this message processor application.

**3. Assumptions**

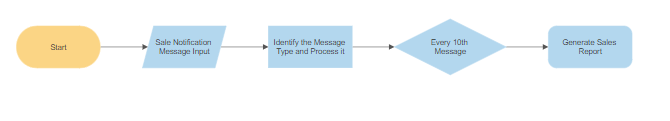
1. All the input notification messages must be within these three defined formats [apple at 10p, 20 sales of apples at 10p each, Add 20p apples] or patterns only. Business information’s like product type, sale count, sale value, adjustment operations (add/multiply/subtract) can be varied.
2. Input messages are passed to the message processing application through java Array lists of strings only, but it can be enhanced to read from files/databases/queue calls using simple utility programs.

**4. Requirements:**

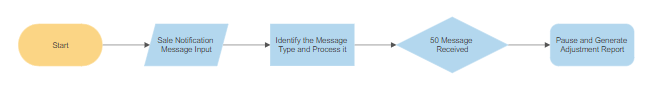
* All sales must be recorded
* All messages must be processed
* After every 10th message received your application should log a report detailing the number of sales of each product and their total value.
* After 50 messages your application should log that it is pausing, stop accepting new messages and log a report of the adjustments that have been made to each sale type while the application was running.

**5.** **Technical Design Diagram**

Generate sales report on every 10th message:



Pause the application and Generate the Adjustment Report if 50 messages are received & processed:



**Interfaces & System Design:**

1. **MessageProcessor.java**

This interface provides the implementation to extract the business information’s from the notification message and processes any of these three operations.

1. If the received message is of Message Type 1 [like apple at 10p]: Then this extracts the information like product type and sale value and considers the default sale count as 1.
2. If the received message is of Message Type 2 [like 20 sales of apples at 10p each]: Then this extracts the information like product type, total sales count and price of each product, after the extraction, we internally calculate the total sale price and proceed further.
3. If the received message is of Message Type 3 [Add 20p apples]: Then this extracts adjustment operation type (add/subtract/multiply) along with product type and sale value to be added to the already processed values.
4. **SaleReport.java**

This interface owns the responsibility to generate the sales report on every 10th message is received and processed successfully.

1. **AdjustmentReport.java**

This interface provides the implementation to generate the adjustment logging report on 50th message received and stops the further message processing.

**6. How to Run/Test?**

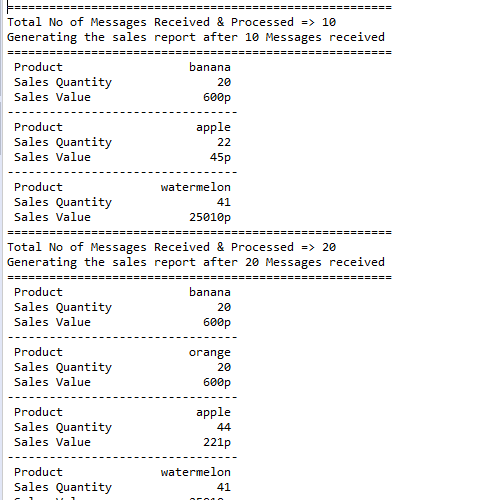
**Prerequisite**:

Java 8 & Maven are required to run this application.

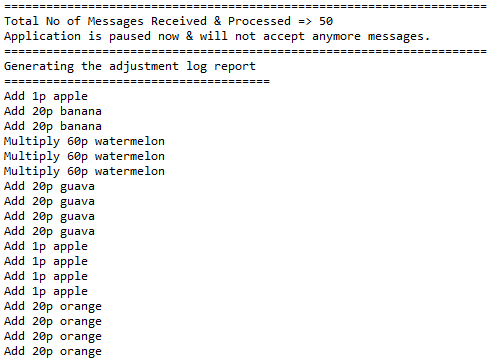
Input data samples are added as part of the test object creator which exists in the application. So, this application can be run directly by performing **maven clean** and **Run as Java Application.**

If any sample data’s needs to be added / modified, then the same can be done in TestObjectCreator.java.

**Generating the sales report Screenshot:**



**Generating the adjustment logging report screenshot:**





**7. Junit & Code Coverage:**

Junit test cases has been written to validate different possible use cases and negative test cases which also helped the application to reach code coverage of 91.7%.

