Software Requirements Specification

Version 3.0

Real-time Cupboard Design

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://[illegible]/

Team 2

Member Name	Member Roll #	Primary Responsibility
M.Mujahid Iqbal	**15L-4105**	**Project Manager**
Asim Fayaz	**15L-4097**	**IV & V Person**
Ahmad Raza	**15L-4041**	**Functional**
Timur Asif	**15L-4012**	**Non-Functional Requirements**
Hassan Jalal	**14L- 4344**	**Risk Analysis and Product Description**

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Table of Contents

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://

Revision History

Name(s)	Date	Reason(s) For Change(s)	Version

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

1. Introduction

Product

A desktop based application for users to design a cupboard design in real time. The product supports both the internal and external design of a cupboard. Drawers and portions or shelves are included in interior and knobs are included in exterior design. Users can customize the design by moving portions of cupboard according to the requirements. The application also gives the estimated cost for the design as well as allows the placement of order directly from application.

Scope

The scope of the project is only limited to one manufacturer i.e. clients can make designs and send an email of those images to the manufacturer who will then actually make cupboards accordingly and will deliver back.

Business Goals

Business goals includes giving relaxation to the clients so that they don't have to always come to the shop and have a proper conversation with the manufacturer but can make or choose designs while sitting in home. Also, since customers are having a portable system of getting cupboards ready so they will prefer this desktop application manufacturer over the ones where they have to visit and tell every detail in person so business of manufacturer having this application will grow.

Document Conventions

- SM stands for Senior Manager.
- DB stands for Database.
- Citation format is APA.
- All heading are in bold.

References

1. Bandakkanavar, R. (2018). Software Requirements Specification document with example - Krazytech. Retrieved from https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database

[1] "Getting started with Javafx" vol. 8, August 1, 2016. [Online]. docs.oracle.com/javase/8/javafx

[2] Kaul, Jeet, "JavaFX — the road ahead", December 18, 2008. [Online]. https://web.archive.org/web/20081217162601/http://blogs.sun.com/meetjeet/entry/javafx_the_road_ahead

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

2. Overall Description

Product Features

User will download this application and then make design of cupboard. After designing the whole cupboard he/she will set budget and delivery time. Then after he is done, the application will send an email to the manufacturer with all images of cupboard, delivery date and budget of the customer. The manufacturer will then respond in email to the customer whether he has accepted his order, rejected or is at stall. Once he has accepted the order and delivered it, the order will be stored in the personal database (DB) of the user's application. So that he can have records of all the orders he has placed till date.

User Classes and Characteristics

There will be just one user i.e. customer because manufacturer will just receive an email from which he has to decide whether to accept it or reject it or put it on hold. So only customer will use this application.

Customer

There will be no sign-up option because manufacturer doesn't care who is his customer, so any customer who has this application will open the application and start making cupboard designs, can then send email and will receive response in the email.

Operating Environment

Hardware requirements

A working PC, laptop and a working internet.

Software Environment

JRE(Java Runtime Environment)

No software requirements and operating system version restrictions.

Design and Implementation Constraints

Hardware limitations: Since application is desktop based so customer should only be able to run it on his laptop or computer.

Interfaces to other Applications: Just email will be sent i.e. email feature that will be used will use Google's Gmail API so a working Gmail account is mandatory.

Language and Communication Protocols are that language used throughout should be English with technical and precise wording and measurements will b made in inches or in meters.

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Security Considerations are that he email will be open to any kind of phishing or hacking but responsible users can avoid that by making contact to the manufacturer so they can confirm their order or can discard one if someone placed an order using their email.

Design Conventions: The desktop application's graphical user interface is as simple and meaningful as possible so any customer can use it easily.

Programming Standards: Java Fx and swing library was used along with java's canvas API to draw objects.

Assumptions and Dependencies

The assumptions made are that:-

- *Each user will use it just for meaningful purpose and won't give other people's emails to create confusion and false orders.*

- *Each user will have a working e-mail id where they can get response regarding their order i.e. the application will not handle the response.*

- *There is only one manufacturer, so all orders will go to just one fixed person whose email will be predefined in the application.*

- *Since response will be given in email so the entire user's orders will be added to db instantly, this will be stored locally because there is no other way for the application to keep track of customer's accepted orders.*

3. Functional Requirements

3.1:

<table>
<tr><td><b>Identifier</b></td><td colspan="2">Design Cupboard</td></tr>
<tr><td><b>Purpose</b></td><td colspan="2">System shall provide a mechanism to draw cupboard design</td></tr>
<tr><td><b>Priority</b></td><td colspan="2">High</td></tr>
<tr><td><b>Actors</b></td><td colspan="2">User</td></tr>
<tr><td><b>Pre-condition(s)</b></td><td colspan="2">Application is running</td></tr>
<tr><td><b>Post-condition(s)</b></td><td colspan="2">Complete design is displayed</td></tr>
<tr><th colspan="3">Typical Course of Action</th></tr>
<tr><th>S#</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td><b>1</b></td><td>User draws the cupboard design</td><td>System show the design</td></tr>
<tr><td><b>2</b></td><td>User clicks on save design</td><td>Prompts for the design name</td></tr>
<tr><td><b>3</b></td><td>Enters the design name to be saved</td><td>Displays “Design Saved Successfully” message</td></tr>
<tr><th colspan="3">Alternate Course of Action 1 (Format is not Correct)</th></tr>
<tr><th>S#</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td><b>1</b></td><td>User clicks on edit the design</td><td>System shows saved designs</td></tr>
<tr><td><b>2</b></td><td>User selects one of the designs</td><td>System opens the saved design and display it.</td></tr>
<tr><td colspan="3">Go to line no 2 in Typical course</td></tr>
</table>

3.2

<table>
<tr><td><b>Identifier</b></td><td colspan="2">Analyze Payment</td></tr>
<tr><td><b>Purpose</b></td><td colspan="2">System shall provide means to allow the user to see the estimated cost</td></tr>
<tr><td><b>Priority</b></td><td colspan="2">High</td></tr>
<tr><td><b>Actors</b></td><td colspan="2">User</td></tr>
<tr><td><b>Pre-condition(s)</b></td><td colspan="2">Design is built and saved</td></tr>
<tr><td><b>Post-condition(s)</b></td><td colspan="2">The final cost is displayed for the design</td></tr>
<tr><td colspan="3"><b>Typical Course of Action</b></td></tr>
<tr><td><b>S#</b></td><td><b>Actor Action</b></td><td><b>System Response</b></td></tr>
<tr><td><b>1</b></td><td>User clicks on Payment Analysis tab</td><td>System opens the amount details window</td></tr>
<tr><td><b>2</b></td><td></td><td>System shows the payment details</td></tr>
<tr><td><b>3</b></td><td>User views the details of cost and proceed</td><td>System shows details in table format with scrolling</td></tr>
</table>

3.3

<table>
<tr><td colspan="2"><b>Identifier</b></td><td>Place Order</td></tr>
<tr><td colspan="2"><b>Purpose</b></td><td>System shall provide method to allow the placement of order of their custom design</td></tr>
<tr><td colspan="2"><b>Priority</b></td><td>High</td></tr>
<tr><td colspan="2"><b>Actors</b></td><td>User and System</td></tr>
<tr><td colspan="2"><b>Pre-condition(s)</b></td><td>Design is built and saved</td></tr>
<tr><td colspan="2"><b>Post-condition(s)</b></td><td>Order is placed and now further correspondence will be done with manufacturer directly.</td></tr>
<tr><td colspan="3"><b>Typical Course of Action</b></td></tr>
<tr><td><b>S#</b></td><td><b>Actor Action</b></td><td><b>System Response</b></td></tr>
<tr><td><b>1</b></td><td>User clicks on Payment Analysis tab</td><td>System opens the amount details window</td></tr>
<tr><td><b>2</b></td><td></td><td>System shows the payment details</td></tr>
<tr><td><b>3</b></td><td>User clicks on order now button</td><td>Prompts for details</td></tr>
<tr><td><b>4</b></td><td>User enters personal details required</td><td>System proceeds with entered inputs</td></tr>
<tr><td><b>5</b></td><td>User clicks on submit</td><td>System sends all the entered details along with pictures of the design to the manufacturer via Email</td></tr>
<tr><td colspan="3"><b>Alternate Course of Action 4 (Format is not Correct)</b></td></tr>
<tr><td><b>1</b></td><td>User clicks on order now button</td><td>System shows “Please enter correct email” message</td></tr>
<tr><td><b>2</b></td><td>User enters correct credentials</td><td>System proceeds accordingly</td></tr>
</table>

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://

3.4

Identifier	View Placed Orders	
Purpose	System shall provide method to display the orders customer has placed	
Priority	Medium	
Actors	User	
Pre-condition(s)	Design is built and Order is placed	
Post-condition(s)	Placed orders cannot be reversed from this method	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	User clicks on View Placed Orders tab	System opens the placed orders window
2		System loads placed order data from database and display
Alternate Course of Action 1 (Format is not Correct)		
1	User clicks on View Placed Orders tab	System opens the placed orders window
2		System displays “No order Placed Yet”

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

4. Nonfunctional Requirements

Performance Requirements

- System shall send order form to Manufacturer within 1 minute given that user has stable internet connection.
- Upon each addition in the design, system should create and display the component just upon clicked.
- While saving order to database and during email sending, system should not stuck user in the interface due to I/O operations etc.

Usability

- Easy to use interface for experts as well as novice users.
- Reversible actions allowed for customization of design and prevention of mistakes

Security Requirements

Authentication

The user must have a valid email address in order for him/her to send or place order to the manufacturer. The email address will be validated according to some regular expressions used for validation. Moreover, exception will be generated by email API when wrong input of email is entered. Also the database should be locally created (using windows authentication) rather than on any other server to make sure orders are for the specific customer only.

5. Other Requirements

Database used will be MySQL database for allowing clients to store their order locally. Database will only contain information of date in which order was placed, dimensions of the cupboard and total price for the design. Besides, the users of this software must be familiar with basic cupboard designing terms like shelves, drawers, knobs etc.

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Appendix A: Glossary

JavaFX

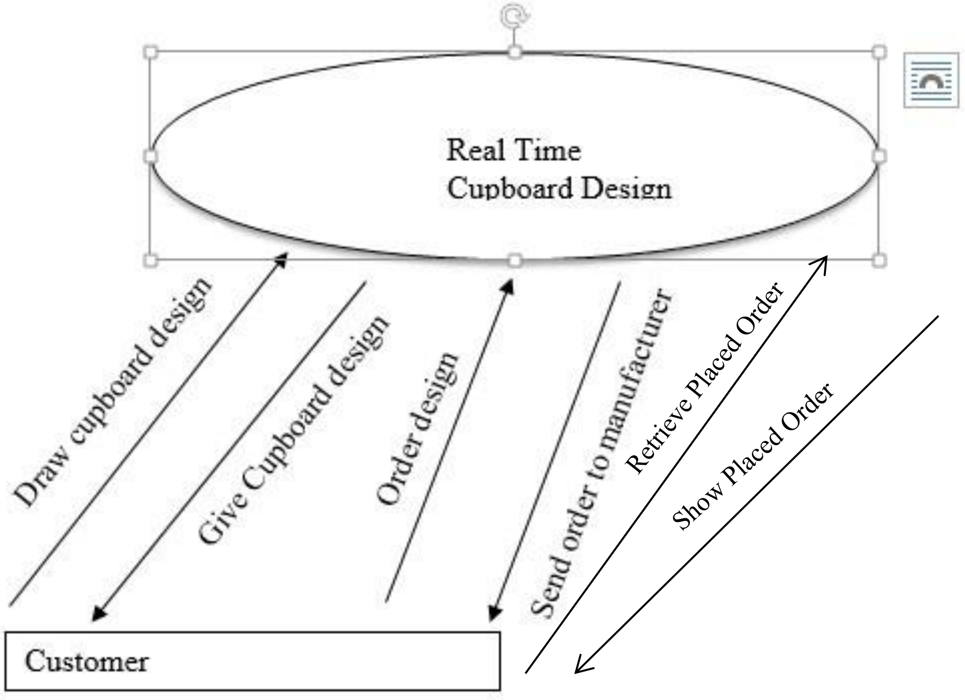
JavaFX [1] is a set of graphics and media packages that enable developers to design, create, test, debug, and deploy rich client applications that operate consistently across diverse platforms. JavaFX have FXML and Scene builder, built-in UI controls and CSS and Canvas API for creating applications. The cross-platform compatibility enables a consistent runtime experience for JavaFX applications, developers and users. Due to this portability it can even be run on other computers (MAC) [2].

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Appendix B: Analysis Models

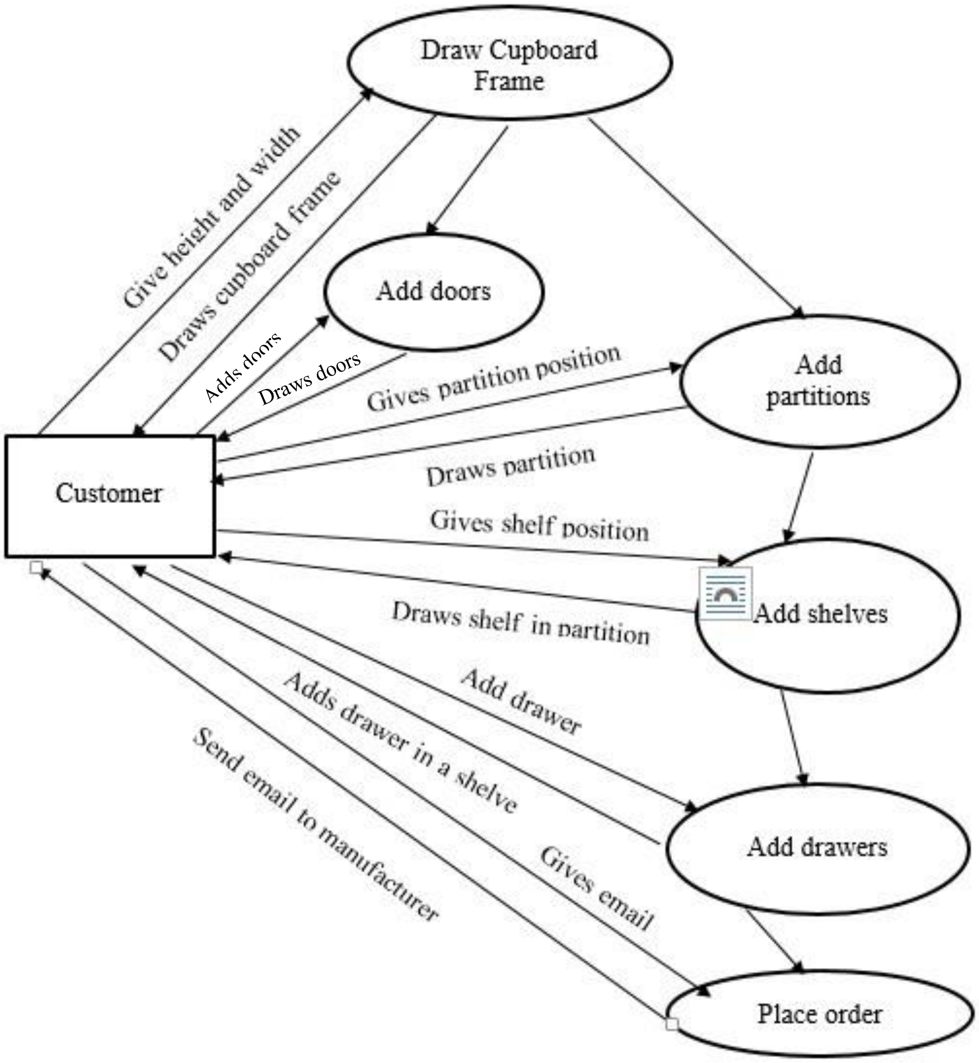
Level 0 DFD:

In level 0 DFD the basic overview of the application is shown. User gives the cupboard design and places the order and manufacturer then contacts him and delivers the order made. Payment method is not in the scope of this project, because it is like OLX app which just connects the buyer (client) and seller(manufacturer) and they can then exchange contact details and work something out then.



Level 1 DFD:

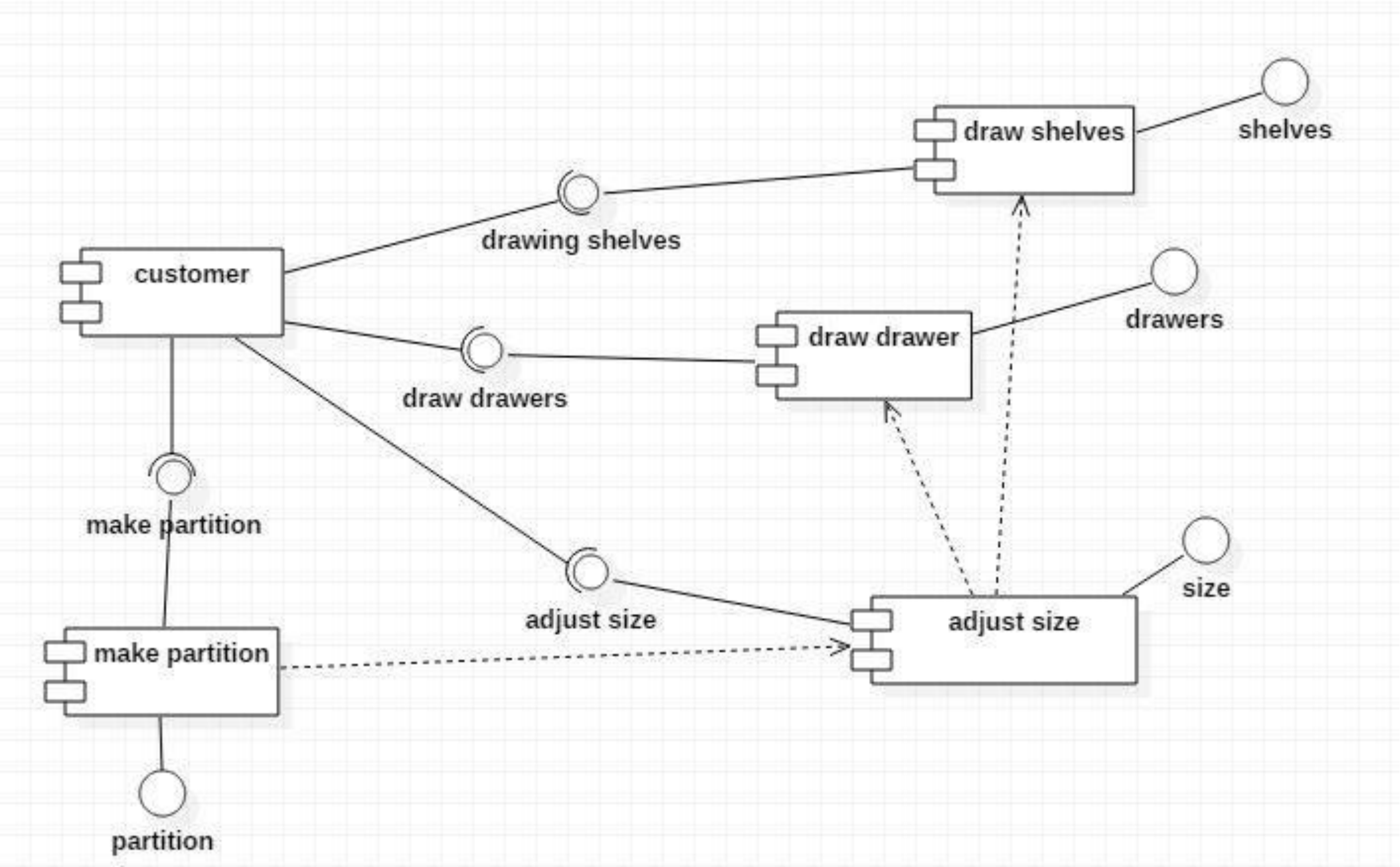
This is level 1 DFD which elaborates in detail the function being done by our application. Customer can add doors, partitions, shelves, drawers and can specify their sizes and can see the design and changes in real time instantly. Also order can be placed and can then be sent over by GMAIL to the manufacturer.



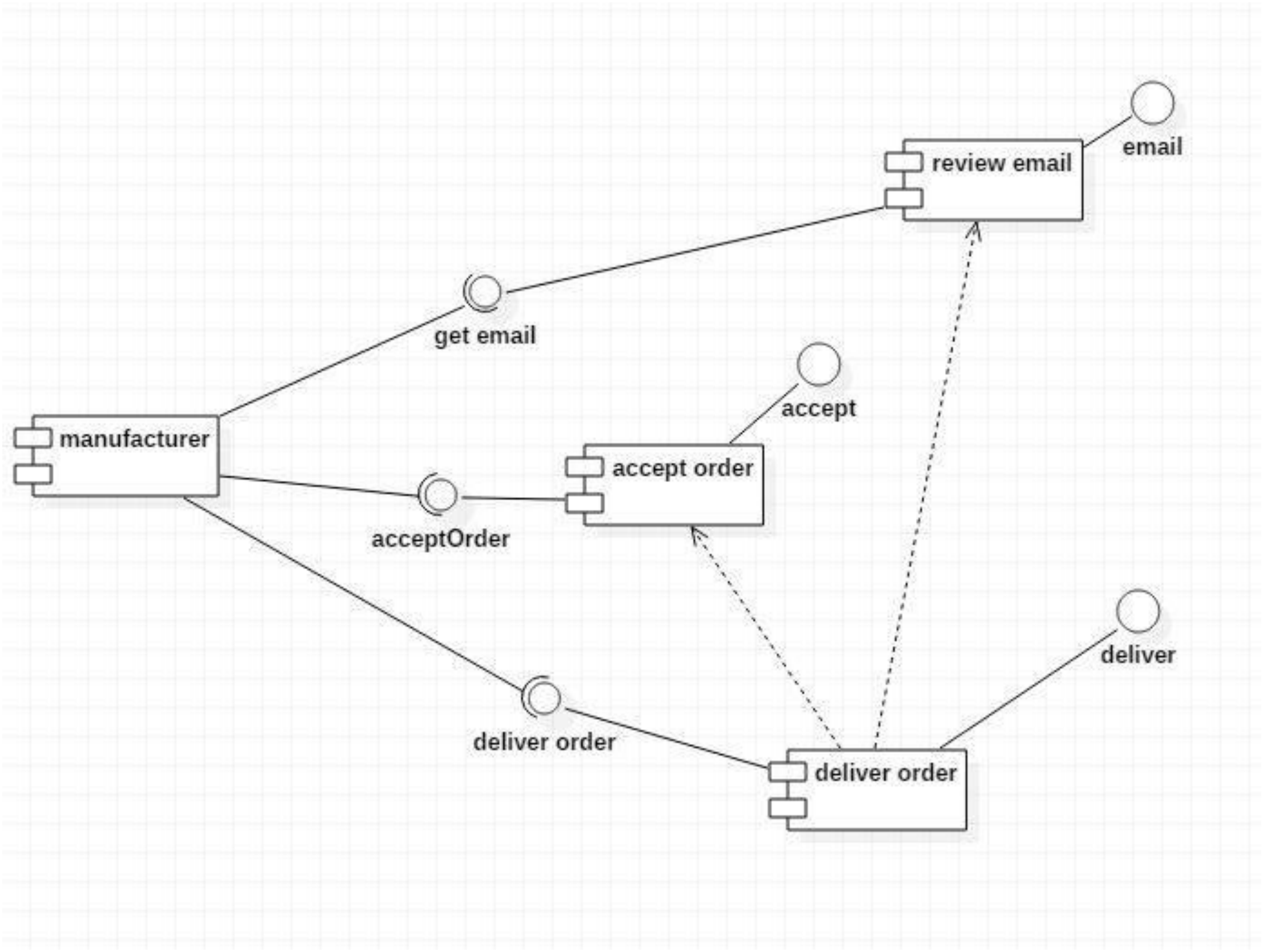
Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Appendix C: Design Models

Component Diagrams



Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://

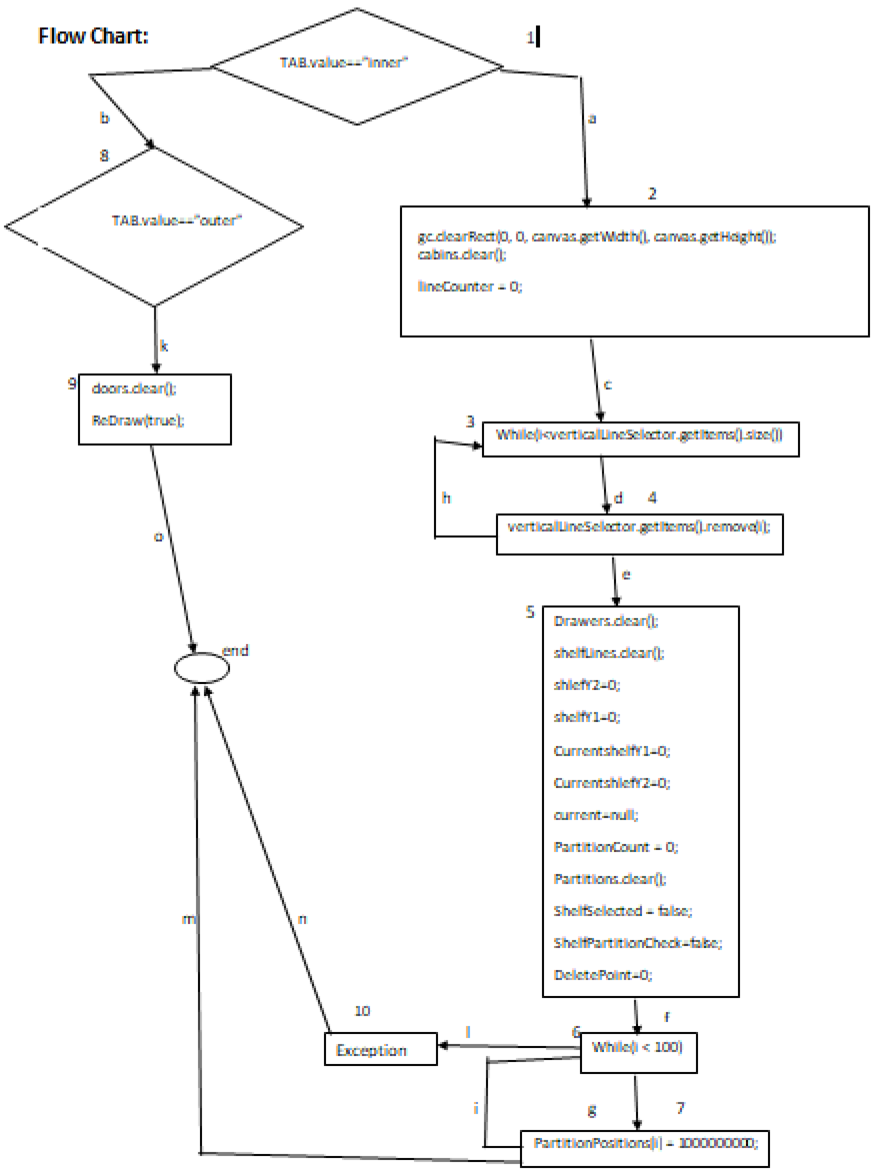


Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Appendix E: Test Cases

1: Controller. Java(Clear()):

<u>Flow Diagram:</u>



Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Test Case ID: Clear

Test Priority (Low/Medium/High): Med

Description: Test the clearing out of inner and outer designs of cupboard

Pre-conditions: Inner and outer design is drawn

Post-conditions: Inner or outer design is cleared out

Path Coverage:

S#	Test Data	Expected Result	Actual Result	Path covered	Status
1	TAB value=”inner”, 1st loop iterations>0, 2nd loop iterations=100	All the partitions,shelves, drawers,cabins and lines must be clear out.	Inner design is cleared out.	1a->2c->3d->4h(>0)->4e->5f->6g->7i(100)->7m	Pass
2	TAB value=”outer”	All doors must be removed.	Doors removed. Correct path choosen	1b->8k->9o	Pass
5	TAB value=”inner”, 1st loop iterations=vertical lines, 2nd loop value=-1	Exception must be thrown.	IndexOutOfBounds Exception thrown. Correct path chosen	1a->2c->3d->4h(>0)->4e->5f->6l->10n	Pass

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://[illegible]

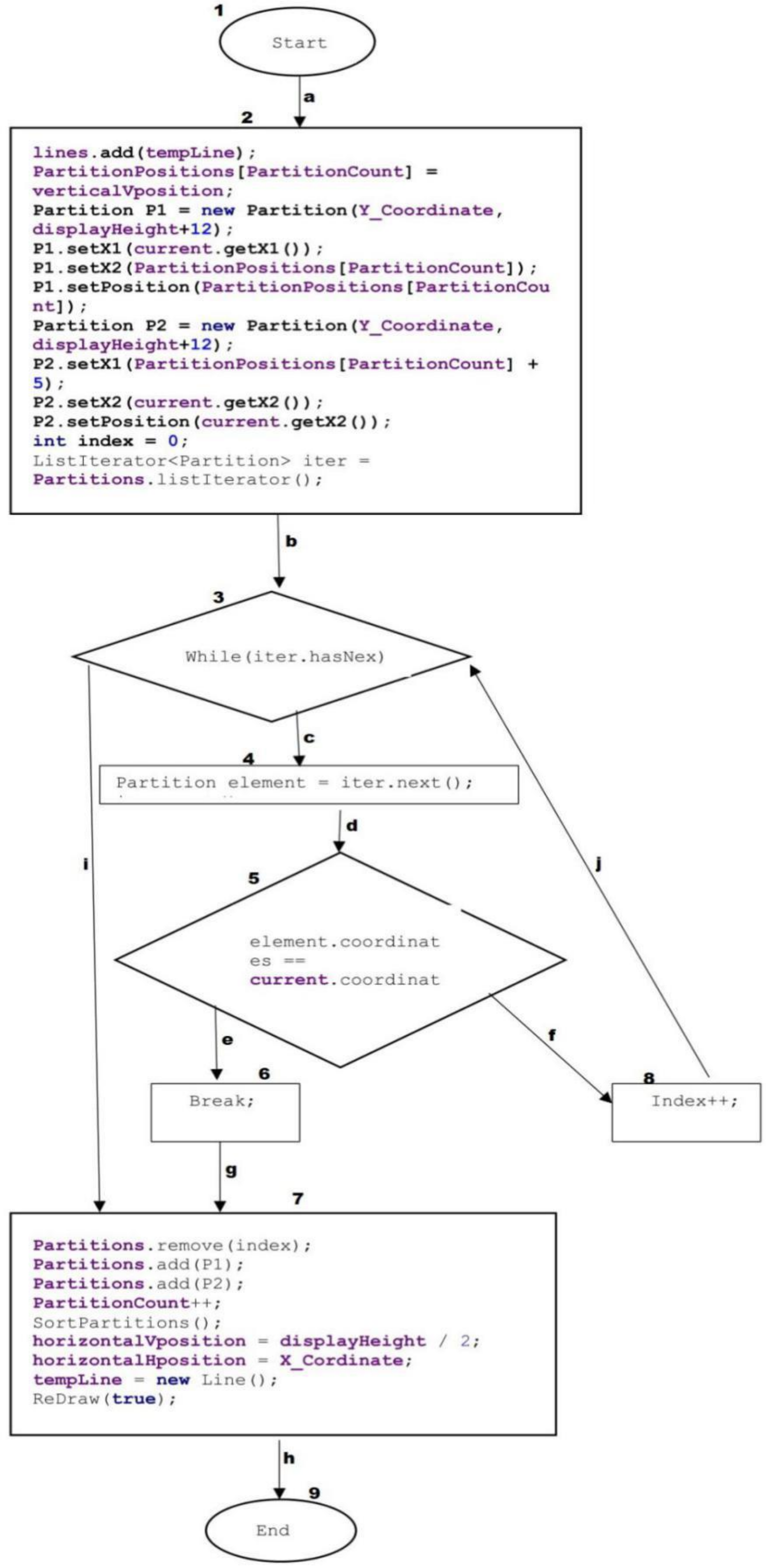
Branch Coverage:

S#	Test Data	Expected Result	Actual Result	Branch path covered	Status (Pass/Fail)
1	TAB value=”inner”, 1st loop iterations=no. of vertical lines drawn, 2nd loop iterations=100	All the partitions,shelves, drawers, cabins and lines must be clear out. 1000000000 should be placed on all array elements of Partitioned Positions	Inner design is cleared out. Correct value on all array elements	1a->2c->3d->4h(no. of lines)->4e->5f->6g->7i(100)->7m	Pass
2	TAB value=”outer”	All doors must be removed.	Doors removed.	1b->8k->9o	Pass
3	TAB value=”inner”,1st loop iterations=0,2nd loop iterations=100	Inner designs must be cleared out. 1000000000 should be placed on all array elements of Partitioned Positions	Inner design contents removed. Array values are correct	1a->2c->3d->4e->5f->6g->7i(100)->7m	Pass
4	TAB value=”inner”,1st loop iterations=0,2nd loop iterations=0	Inner designs must be cleared out.	Inner contents cleared out.	1a->2c->3d->4e->5f->6g->7m	Pass
5	TAB value=”inner”, 1st loop iterations=vertical lines, 2nd loop value=-1	Exception must be thrown	IndexOutOfBounds Exception thrown	1a->2c->3d->4h(no. of lines)->4e->5f->6l->10n	Pass
6	TAB value=”inner”, 1st loop iterations=vertical lines, 2nd loop value=0	All the partitions,shelfs,drawers,cabins and lines must be clear out.	Inner design is cleared out.	1a->2c->3d->4h(no. of lines)->4e->5f->6g->7m	Pass

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

2: Controller. Java(SavePartition()):

Flow Diagram:



```
lines.add(tempLine);
PartitionPositions[PartitionCount] =
verticalVposition;
Partition P1 = new Partition(Y_Coordinate,
displayHeight+12);
P1.setX1(current.getX1());
P1.setX2(PartitionPositions[PartitionCount]);
P1.setPosition(PartitionPositions[PartitionCou
nt]);
Partition P2 = new Partition(Y_Coordinate,
displayHeight+12);
P2.setX1(PartitionPositions[PartitionCount] +
5);
P2.setX2(current.getX2());
P2.setPosition(current.getX2());
int index = 0;
ListIterator<Partition> iter =
Partitions.listIterator();
```

```
Partitions.remove(index);
Partitions.add(P1);
Partitions.add(P2);
PartitionCount++;
SortPartitions();
horizontalVposition = displayHeight / 2;
horizontalHposition = X_Cordinate;
tempLine = new Line();
ReDraw(true);
```

is

Test Case ID: Save_Partion

Test Priority (Low/Medium/High): Medium

Description: Testing of partition being saved

Pre-conditions: A vertical partition is drawn

Post-conditions: Vertical partition saved permanently

Total statements = 28

Total Paths = 03

Cyclomatic Complexity:

E-N+2P = 10-9+2*11

= 1+22 = 23

Statement Coverage:

=> 28/28 = 100%

Path Coverage:

S#	Test Data	Expected Result	Actual Result	Path covered	Status (Pass/Fail)
1	Iter.hasnext()==false	Partition saved to any location	Partition saved to 0th position	1a->2b->3i->h9	Pass
2	Iter.hasnext()==”true”, element.coordinates==current.coordiantes	Partition saved 1st position	Partition saved to 0th position	1b->2b->3c->4d->5e->6g->7h->9	Fail
3	Iter.count==partitions.size”, current==lastElement	Partition saved to last location.	Partition saved to position of last index	1b->2b->3c->4d->5f->8i->3c->4d->5e->6g->7h->9	Pass

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://

Branch Coverage:

S#	Test Data	Expected Result	Actual Result	Branch path covered	Status (Pass/Fail)
1	Iter.hasnext()==”true”, element.coordinates==current.coordiantes(at 2nd itertation)	Partition saved at 2nd position	Partition saved to 1th position	1b->2b->3c->4d->5f->8i->3c->4d->5e->6g->7h->9	Pass
2	Iter.hasnext()==”true”, element.coordinates!=current.coordiantes(always false)	Partition must be saved to maximum available location.	Partition saved to position of last index+1	1b->2b->3c->4d->5f->8i->3i->7h->9	Pass
3	Iter.hasnext()==”true”, element.coordinates==current.coordiantes(at first itertation)	Partition saved at 1st position	Partition saved to 0th position	1b->2b->3c->4d->5e->6g->7h->9	Fail
4	Iter.hasnext()==”true”, element.coordinates==current.coordiantes	Partition must be saved to non-zero locatoin.	Partition saved to 1st position	1b->2b->3c->4d->5e->6g->7h->9	Pass
5	Iter.hasnext()==”true”, element.coordinates==current.coordiantes(anywhere between max and min)	Partition must be saved to any location	Partition saved to position of index between 0 and maximum	1b->2b->3c->4d->5f->8i->3i->7h->9	Pass
6	Iter.count==partitions.size”, current==lastElement	Partition must be saved to maximum available location.	Partition saved to position of last index	1b->2b->3c->4d->5f->8i->3c->4d->5e->6g->7h->9	Pass

3: Order. Java (Space to File()):

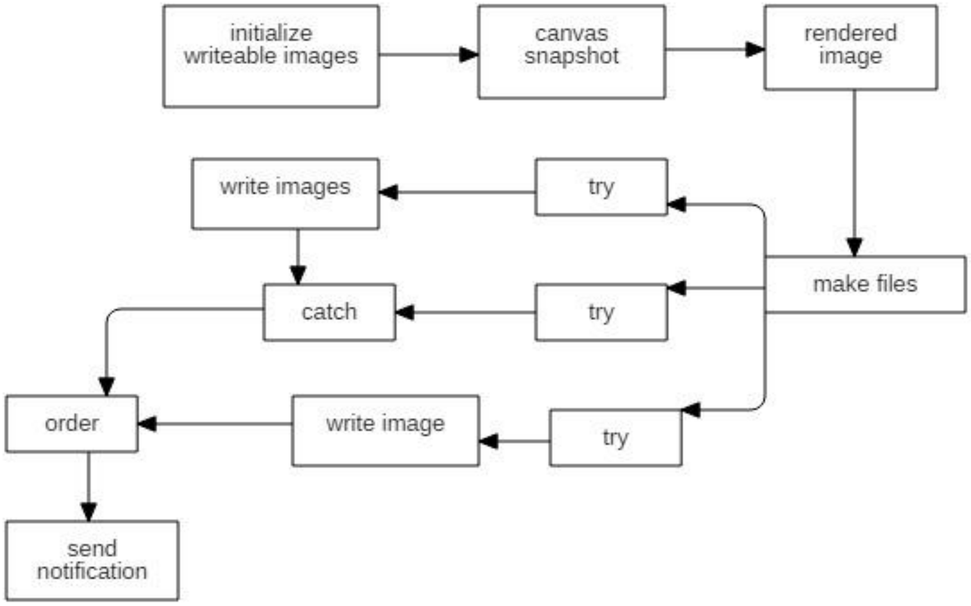
Total statements = 26

Total Paths = 03

Total Brakes = 02

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Central Flow Graph:



Test Case ID: SpaceToFile

Test Priority (Low/Medium/High): High

Description Testing of 2D space drawings to images

Pre-conditions: Both inner and outer designs are created

Post-conditions: Both designs saved to images

Cyclomatic Complexity:

F-N+2P = 13-12+2*11

= 13-12+22 = 23

Statement Coverage:

=> 26/26 = 100%

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Shortest Path To Cover All Nodes:

1->2->3->4->6->10->9->11->12

Test Cases:

Six test cases will be needed to cover all branches.

Test Case ID	Actual Output	Expected Output	Status
1	Record Gather	Record Gather	Pass
2	Error Occured	Error Occured	Pass
3	Error in Class File	Error in Class File	Pass
4	Buffer Error	Buffer Error	Pass
5	Record Gather	Record Gather	Pass
6	Error	Error	Pass

Boundary Case Testing:

ID	Boundary Value	Actual Output	Expected Output	Status
1	Count = 9	3 lines read	3 lines read	Pass
2	Count = 0	Nothing read	Nothing read	Pass
3	Count-> infinity	All data read	All data read	Pass

Loop Case Testing:

ID	Loop Scenario	Actual Output	Expected Output	Status
1	Skip Loop Entirely	Nothing read	Nothing read	Pass
2	Take one pass	1 line read	1 line read	Pass
3	Take multiple passes	Q lines read	Q lines read	Pass
4	Infinite passes	Whole file read	Whole file read	Pass

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http:// [illegible]

4: Store Db. Java(getRecord()):

Test Case ID: Get_Record

Test Priority (Low/Medium/High): High

Description Testing of getting previously saved records

Pre-conditions: Some order's information stored to database

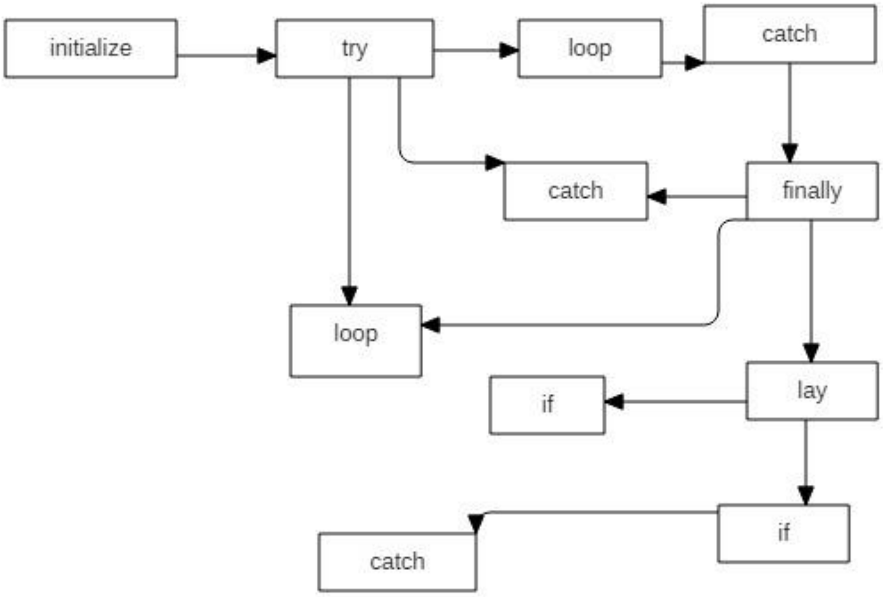
Post-conditions: Previously stored records shown to screen

Total statements = 26

Total Paths = 03

Total Brakes = 02

Central Flow Graph:



Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Cyclomatic Complexity:

G-N+2P = 12-11+2*10

= 12-11+20 = 21

Statement Coverage:

=> 17/22 = 77.97%

Shortest Path To Cover All Nodes:

1>2->3->6->7->8->9->11

Branch Coverage:

Path1 : 1>2->3->6->7->8->9->11

Path2 : 1>2->4->7->8->10

Path3 : 1>2->5->7->8->10

Path4 : 1>2->4->7->8->9->11

Path5 : 1>2->5->7->8->9->11

Path6 : 1>2->3->6->7->8->9->10

Branch Coverage:

Path1 : 1>2->3->4->6->10->9->11->12

Path2 : 1>2->3->4->8->9->11->12

Path3 : 1>2->3->4->7->8->11->12

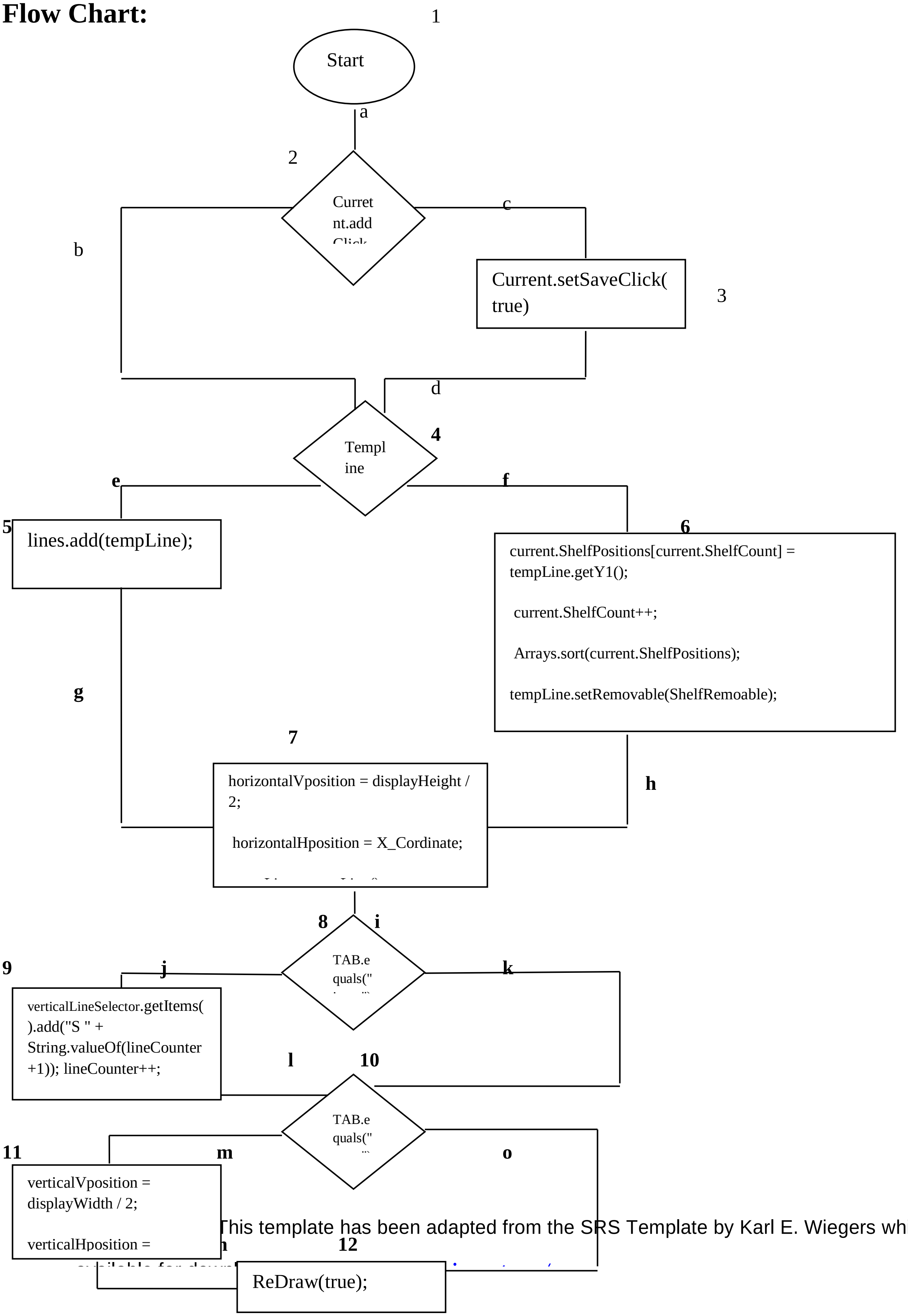
Test Case ID	Actual Output	Expected Output	Status
1	Image Written	Image Written	Pass
2	Error	Error	Pass

As there are no loops so we there won't be any boundary or loop case testing.

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://

5: Controller. Java(SaveLine()):

Flow Chart:



Test Case ID: Save Line

Test Priority (Low/Medium/High): High

Description: Testing of line being saved

Pre-conditions: Line is drawn

Post-conditions: Line is saved

Path Coverage:

S#	Test Data	Expected Result	Actual Result	Path covered	Status (Pass/Fail)
1	Templine.getY1()=template.getY2()	Shelf is being saved	Shelf is being saved at current shelf counter	1a-2c-3d-4f-6h-7i-8j-9l-10o-12	Pass
2	TAB.equals=”outer”	vertPos=displayWidth/2	vertPos=displayWidth/2	1q-2c-3d-4f-6h-7i-8j-9l-10m-11n-12	Pass
3	Templine.getY1()!=template.getY2()	Add Templine	Add Templine	1a-2c-3d-4c-5g-7i-8j-9l-10o-12	Pass

Branch Coverage:

S#	Test Data	Expected Result	Actual Result	Path covered	Status (Pass/Fail)
1	Templine.getY1()=template.getY2()	Shelf is being saved	Shelf is being saved at current shelf counter	1a-2c-3d-4f-6h-7i-8j-9l-10o-12	Pass
2	Templine.getY1()!=template.getY2()	Add Templine	Add Templine	1a-2c-3d-4c-5g-7i-8j-9l-10o-12	Pass

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Appendix F: IV & V Report

IV & V Resource

Name Roll # Signature

S#	Defect Description	Origin Stage	Status	Fix Time	
				Hours	Minutes
1	Spelling mistakes throughout	Phase 2	Fixed		30
2	Use case diagram had missing sections	Phase 1	Fixed	1	0
3	Nonfunctional requirements had ambiguity	Phase 2	Fixed	1	0

Table 3: List of non-trivial defects

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Appendix G: Risk Report

[1]Project Risks

Risk Description	Impact (1 – 10)	Probability (0 – 1)	[2]Risk Exposure	Weeks Active	Mitigation Strategy
Time Management risk	10	0.9	9.0	8	assigning deadlines to every team member and ever module of project
work schedule risk	8	0.8	6.4	1	Pre plan the flow of tasks. decide which tasks could be performed in parallel and which are dependent on other tasks.
Quality maintenance risk	7	0.5	3.5	8	checking the quality of every document and module in parallel with development
Design risk	5	0.35	1.75	3	meeting the client and discuss the design and proceed accordingly

Usability risk	5	0.3	1.5	1	designer will work to make user friendly design which is both efficient and easy to interact with
integration risk	4	2.5	1	8	following standard conventions to make module integration easy

[1] Risks should be sorted in descending order of risk exposure.

[2] Risk Exposure = Risk Impact x Risk Probability

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://

Appendix H: Activity Timesheet

Activity	Time	
	Hours	Minutes
Requirements Engineering	14	0
Analysis and Design	24	
Implementation	106	
Testing	6	30
Deployment		
Project Management		
IV & V	2	30

Project Manager

Muhammad Mujahid 15L-4105

Name Roll # Signature

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://www.processimpact.com/

Important Note: This template has been adapted from the SRS Template by Karl E. Wiegers which is available for download at http://[illegible]