Decision Lines System Design

Overview, Glossary, Use Cases, Story Boards, & Class Diagrams

**Authors:**

**Trevor Hodde**

**James Forkey**

**Wei Chen**

**Hang Cai**

**Martti Peltola**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

*(This page intentionally left blank)*

**TABLE of CONTENTS**

REVISION HISTORY 4

1. Introduction 5

1.1 Scope 5

1.2 Glossary 5

2. Functional Overview 5

3. Principles of Operation 5

4. Use Cases 5

4.1.1 Use Cases derived from Scenario 2.4.1.1 5

4.1.2 Use Cases derived from Scenario 2.4.1.2 5

4.1.3 Use Cases derived from Scenario 2.4.1.3 “Moderator creates closed design line event” 5

4.1.3.1 Use Case: Moderator Creates Closed Design Line Event 6

4.1.3.2 Use Case: Add Valid User RoundRobin Edge 6

4.1.3.3 Use Case: Attempt Invalid Edge Value 6

4.1.3.4 Use Case: Attempt Invalid Turn At RoundRobin Edge 7

4.1.3.5 Use Case: Attempt extra turn choosing Edge 7

4.1.3.6 Use Case: UserEnters Invalid EventID 8

4.1.3.7 Use Case: Add Valid User Async Edge 9

4.1.4 Use Cases derived from Scenario 2.4.1.4 10

4.1.5 Use Cases derived from Scenario 2.4.1.5 10

4.1.6 Use Cases derived from Scenario 2.4.1.6 10

5. STORY Board 11

5.1 Login Screens 11

5.2 Moderator 12

5.2.1 Moderator creates Closed event 12

5.3 User 14

5.3.1 User Entering Open Event Choice 14

5.4 Administrator 16

5.4.1 Event Management 16

5.4.2 Report Generation 16

6. CLASS DIAGRAM 18

6.1 CLASS DIAGRAMS For USE CASES DERIVED FROM SCENARIO 2.4.1.1 18

6.2 CLASS DIAGRAMS For USE CASES DERIVED FROM SCENARIO 2.4.1.2 18

6.3 CLASS DIAGRAMS For USE CASES DERIVED FROM SCENARIO 2.4.1.3 18

6.3.1 CLASS DIAGRAM (ECB) For Moderator Creates Closed Event 18

6.4 CLASS DIAGRAMS For USE CASES DERIVED FROM SCENARIO 2.4.1.5 18

6.5 CLASS DIAGRAM 1 19

6.6 CLASS DIAGRAM 2 19

6.7 CLASS DIAGRAM 3 19

**REVISION HISTORY**

| **Rev** | **Date/By** | **Release #** | **Description** |
| --- | --- | --- | --- |
| Draft 1 | 09/21/2012/  M. Peltola | -none- | Initial Template. |
| Draft 2 | 09/22/2012/M. Peltola |  | Moved existring use cases for scenario 2.4.1.3 into this document. Raw UI Mockups for storyboard, ECB class diagram for single use case. |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Scope

This document is derived from the **Requirements Analysis Document (RAD), Decision Lines [Draft: 9-4-2012]**

## Glossary

Edge Height chosen by user for a given choice.

Moderator The creator and owner of a Decision Lines event

User Member of group with choice privileges for a given event

Administrator Person with granted special maintenance privileges via a restricted UI interface

Event A specific question and choices created by a moderator.

EventID A unique identifier assigned to a decision line event..

# Functional Overview

This section identifies the functions covered by this document.

# Principles of Operation

The purpose of this project is to add

# Use Cases

Use Cases derived from Scenario 2.4.1.3 “Moderator creates closed design line event”

### Use Cases derived from Scenario 2.4.1.1

### Use Cases derived from Scenario 2.4.1.2

### Use Cases derived from Scenario 2.4.1.3 “Moderator creates closed design line event”

Rationale:

The title describes the scenario as covering a moderator creating a closed design lines event. However, the scenario also covers the actor instance playing the role of both moderator and user. This scenario will derive distinct use cases for both the moderator and user.

The scenario does not cover the possibility of creating either a round robin or asynch event. I include this choice as a part of the step, an alternate approach could require a use case for each mode.

I only extracted a single moderator use case for this scenario, and 6 for the user.

#### Use Case: Moderator Creates Closed Design Line Event

Participating Actor: Moderator

Flow of Events:

1. Moderator requests a new decision lines event

2. DLS presents Create Decision Lines Event Dialog

3. Moderator requests event to be closed

4. DLS modifies dialog to display only input controls relevant for Closed Event

5. Moderator enters question for event, the number of times a user can select a choice, the number of choices, text for the choices, user entry mode(round robin/asynch), and presses submit.

Entry Criteria:DLS app has already been launched

Exit Criteria: EventID returned to user

Note: I combined all the user option entries (step 5) since there may not be an even step response after each option is entered

Note: The Entry and Exit criteria could be additional flow of event steps, but these seem better suited as ‘criteria’

#### Use Case: Add Valid User RoundRobin Edge

Participating Actor: User

Flow of Events:

1. User requests adding new decision to an existing closed decision lines event

2. DLS asks for event ID

3. User enters eventID

4. DLS informs user he is allowed a choice at this time

5. User makes his choice and submits choice

Entry Criteria:DLS app has already been launched

Exit Criteria: DLS acknowledges choice has been accepted

#### Use Case: Attempt Invalid Edge Value

Participating Actor: User

Flow of Events:

1. User requests adding new decision to an existing closed decision lines event

2. DLS asks for event ID

3. User enters eventID

4. DLS informs user he is allowed a choice at this time

5. User makes his choice and submits choice

6 DLS informs user his choice value is too close to a previous choice value

Entry Criteria:DLS app has already been launched

Exit Criteria: DLS acknowledges choice has been accepted

#### Use Case: Attempt Invalid Turn At RoundRobin Edge

Participating Actor: User

Flow of Events:

1. User requests adding new decision to an existing closed decision lines event

2. DLS asks for event ID

3. User enters eventID

4. DLS informs user he is not allowed a new choice at this time

Entry Criteria:DLS app has already been launched

Exit Criteria: DLS returns to initial screen

#### Use Case: Attempt extra turn choosing Edge

Participating Actor: User

Flow of Events:

1. User requests adding new decision to an existing closed decision lines event

2. DLS asks for event ID

3. User enters eventID

4. DLS informs user he has already entered all his allowed choices

Entry Criteria:DLS app has already been launched

Exit Criteria: DLS returns to initial screen

#### Use Case: UserEnters Invalid EventID

Participating Actor: User

Flow of Events:

1. User requests adding new decision to an existing closed decision lines event

2. DLS asks for event ID

3. User enters eventID

4. DLS informs user his eventID is invalid

Entry Criteria:DLS app has already been launched

Exit Criteria: DLS returns to initial screen

#### Use Case: Add Valid User Async Edge

Participating Actor: User

Flow of Events:

1. User requests adding new decision to an existing async closed decision lines event

2. DLS asks for event ID

3. User enters eventID

4. DLS informs user he is allowed a choice at this time

5. User makes his choice and submits choice

Entry Criteria:DLS app has already been launched

Exit Criteria: DLS acknowledges choice has been accepted, has x remaining turns, and asks if user wishes another choice

Note: no async use case for turns used up, same as earlier use case

### Use Cases derived from Scenario 2.4.1.4

### Use Cases derived from Scenario 2.4.1.5

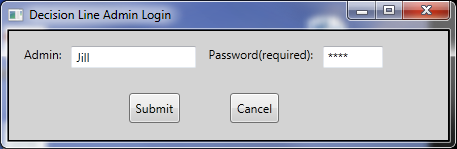
### Use Cases derived from Scenario 2.4.1.6

# STORY Board

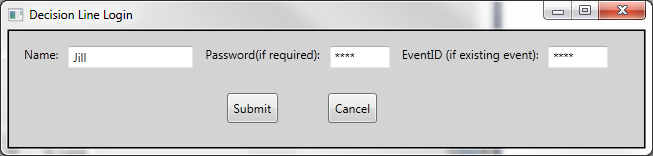
**How to Read this Section**

## Login Screens

Here is an Admin Login Screen



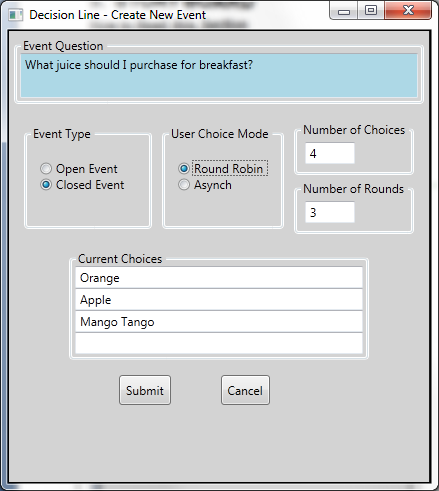
Moderator and user login. If EventID is left blank, it assumes you are logging in as a moderator who pnas to create a new event. If an eventID of an existing eventID is entered, and the name is the moderator, the event editing screen is entered. If you are a user, you go to a user oriented screen (add a choice for an open event, or select a choice for a closed event. You are rejected if the eventID is not valid for your name. If you have no further choices, and no decision has been reached, you are informed that you have entered all your choices, and the event is waiting for the choices of others. If the decision has been made, you are informed of the choice.



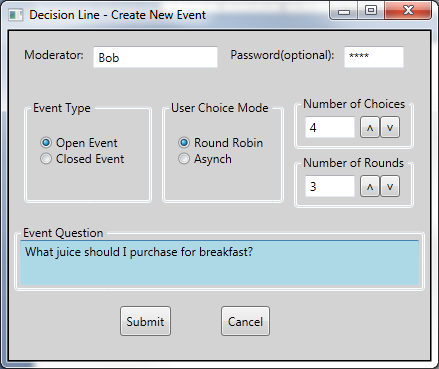
## Moderator

### Moderator creates Closed event

Here is the closed event, and the moderator has the controls to enter the choices (one more remains). We can disable the submit button until 4 entries are input (preferred) or if the moderator attempts to submit with 3 entries or less, we put up an error dialog (more work for us and end user, not preferred)



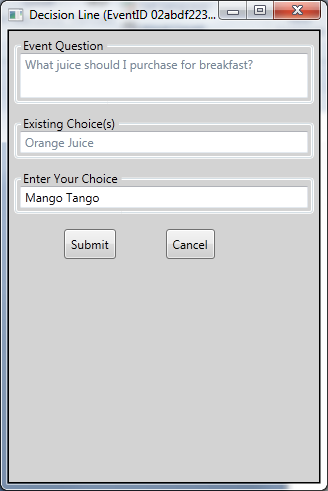
Here is the screen shot for an Open Event, note that there is no control to enter choices, those will be entered by users.

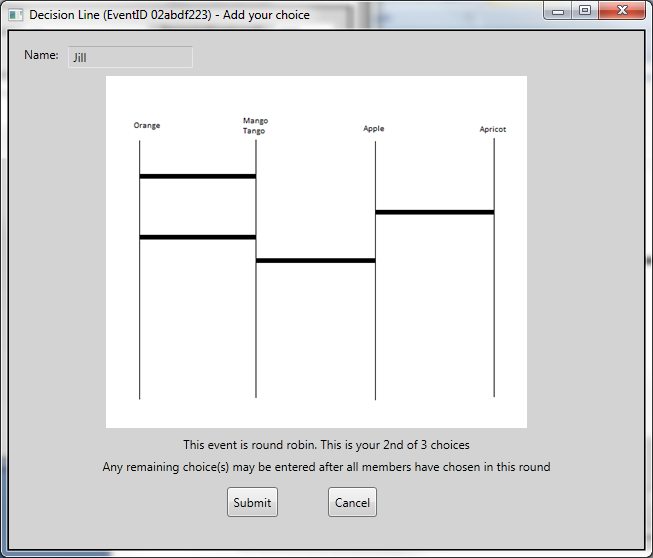


## User

### User Entering Open Event Choice

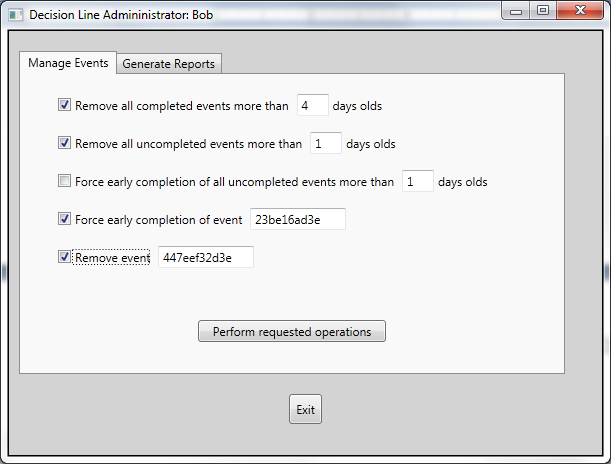
In the figure below, the user is entering her choice in an open event. She is the second person to enter choice information (Orange Juice has been entered by a previous user).



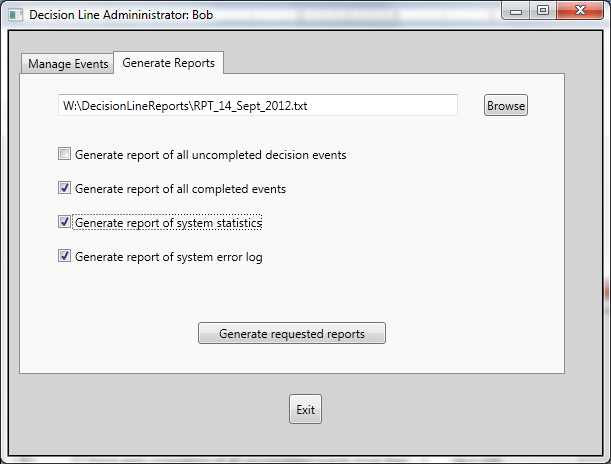


## Administrator

### Event Management



### Report Generation



# CLASS DIAGRAM

**How to Read this Section**

## CLASS DIAGRAMS For USE CASES DERIVED FROM SCENARIO 2.4.1.1

## CLASS DIAGRAMS For USE CASES DERIVED FROM SCENARIO 2.4.1.2

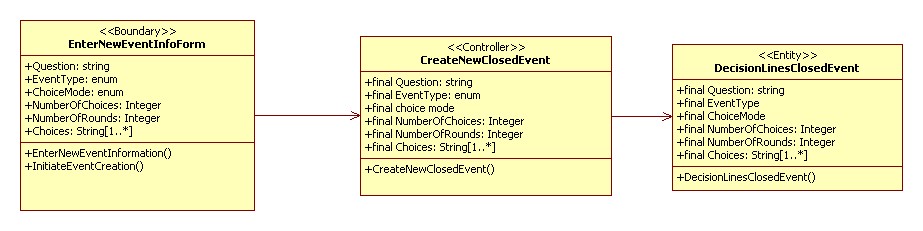
## CLASS DIAGRAMS For USE CASES DERIVED FROM SCENARIO 2.4.1.3

### CLASS DIAGRAM (ECB) For Moderator Creates Closed Event

The Boundary Object, EnterNewEventInfoForm, provides an attribute for each data item the moderator can modify in the creation of a new closed event. It provides 2 operations, a constructor to initialize the object, and InitiateEventCreation, which sends a message to the controller object, along with the chosen options.

The controller object, CreateNewClosedEvent realizes the primary use case of the scenario, It provides momentary (does this violate the goal of being stateless?) attributes to store choices provided by the boundary object. Besides its constructor, it provides one other operation, which creates a new closed event object from the attriubutes supplied from the boundary object.

The entity object provides attributes which hold the states of the new event entity object. The entity object provides a signle operation, its contructor, in addition to read only access to its attributes.



## CLASS DIAGRAMS For USE CASES DERIVED FROM SCENARIO 2.4.1.5

## CLASS DIAGRAM 1

**How to Read this Section**

## CLASS DIAGRAM 2

**How to Read this Section**

## CLASS DIAGRAM 3

**How to Read this Section**