UML HW 3

Andres Namm

Contents

[Assignement Specifications 2](#_Toc468176509)

[Part I: Before the Requirement Change 3](#_Toc468176510)

[1. Transform the analysis use case documentation of use case “Register” to design use case documentation. 3](#_Toc468176511)

[2. Model object interaction and behaviors for the Design Use Case “Register”. 6](#_Toc468176512)

[i. Identify and categorize design classes; 6](#_Toc468176513)

[ii. Identify (additional) attributes; 6](#_Toc468176514)

[iii. Identify behaviors and responsibilities; 6](#_Toc468176515)

[3. Communication diagram(s) for the Design Use Case “Register”. 11](#_Toc468176516)

[4. Update the class diagram from the one appeared in Assignment 1 sample solution so that it only reflects (i) the changes of object classes relevant to the Design Use Case “Register”, and (ii) design class identified above in Step 2. 13](#_Toc468176517)

[Part II: After the Requirement Change 14](#_Toc468176518)

[1. Transform the analysis use case documentation of use case “Register” to design use case documentation. 14](#_Toc468176519)

[2. Model object interaction and behaviors for the Design Use Case “Register”. 14](#_Toc468176520)

[i. Identify and categorize design classes; 14](#_Toc468176521)

[ii. Identify (additional) attributes; 14](#_Toc468176522)

[iii. Identify behaviors and responsibilities; 15](#_Toc468176523)

[3. Communication diagram(s) for the Design Use Case “Register”. 16](#_Toc468176524)

[(To ease student workload, low-level interactions with sequence diagram(s) and state machine(s) need not be prepared for this assignment.) 16](#_Toc468176525)

[4. Update the class diagram from the one appeared in Assignment 1 sample solution so that it only reflects (i) the changes of object classes relevant to the Design Use Case “Register”, and (ii) design class identified above in Step 2. 16](#_Toc468176526)

# Assignement Specifications

**Background**

A team of systems analysts have worked out the analysis artifacts of the TSE system. You are assigned to work on the design phase accordingly.

You need complete the tasks defined in the next section. Please refer to the sample solution of Assignment 1 shown on Moodle course webpage as their analysis artifacts passed to you. Your design model should reflect the requirements in Section 2 and at the same time be in accordance with the sample analysis model as in the sample solution of Assignment 1.

**Tasks and Deliverables**

Please conduct the following standard steps for the part of the TSE system

**Part I: Before the Requirement Change**

1. Transform the analysis use case documentation of use case “Register” to design use case documentation.

2. Model object interaction and behaviors for the Design Use Case “Register”. i. Identify and categorize design classes;

ii. Identify (additional) attributes;

iii. Identify behaviors and responsibilities;

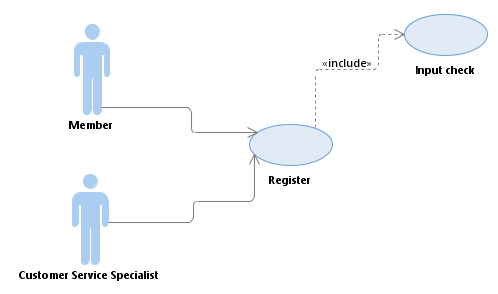
3. Communication diagram(s) for the Design Use Case “Register”.

(To ease student workload, low-level interactions with sequence diagram(s) and state machine(s) need not be prepared for this assignment.)

4. Update the class diagram from the one appeared in Assignment 1 sample solution so that it *only* reflects (i) the changes of object classes relevant to the Design Use Case “Register”, and (ii) design class identified above in Step 2.

# Part I: Before the Requirement Change

## 1. Transform the analysis use case documentation of use case “Register” to design use case documentation.



**xxxx - Behaviours**

**xxxx - Possible additional classes**

**xxxx – Classes identified in the analysis phase**

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Register | |
| **Actor(s):** | Member | |
| **Description:** | This use case describes the membership registration process | |
| **Reference:** | TS-1 | |
| **Typical Course of Events:** | **Actor Action** | **System Response** |
| The main window is currently displayed  on the screen waiting for the customer  to select a register option.  **Step 1**: Initiate the use case when a new customer clicks on the button “*register*” on the main window, which is located on the touchscreen machine at TSE cinema lobby. Windows are shown on a browser interface, so it would be easy to transform it to web format.  **Step 3**: Input the personal information and click submit.  **Step 8:** Pick the printed paper up with the registration key up and go to the card pickup counter and hand the registration slip to the customer service specialist  **Step 9:** Customer service specialist receives the slip and inputs it in the current registrations window textbox and clicks submit.  **Step 11:** Customer service specialist clicks confirm.  **Step 16:** Receive the printed membership card from the printer and hand it over to customer. | **Step 2**: Display a registration form window, where there are textboxes with labels: name, address, email, birthday , (in DD/MM/YYYY format) password, password repeat and Hong Kong mobile phone number (All Marked compulsory). On the display, there is the submit button on bottom part of the screen.    **Step 4:** Invoke the use case Input Check  **Step 5:**  Generate randomly registration ID. Store the data together with the Registration ID for 30 minutes to the temporary registrations list  **Step 6:** Print out a randomly generated registration id.  **Step 7:** Display a notification window to the customer*: “To confirm your account creation, please pick the printed registration ID and go to the counter to pick up your membership card*”.  **Step 10:** display a windowwhich shows in a list the customer inserted info. With a confirm button  **Step 12:**  Generate membership number in format TSENTITY(number of previous registrations)+1.  , then store the registration info together with the membership number to the database.  **Step 13:** Send a message to the card printer **to** Assign the card to the membership ID by RFID tag and print out the plastic card.  **Step 14:** Send Entitymail to the customer with text: Dear Mr/Mrs <Name> Your account has been created. Your membership number is <Membership number> and your password is <password>.  **Step 15:**  Display successful account creation window for 30 seconds or until screen is klicked and then resume to .current registration window  Conclude the use case when a membership number and a password is assigned to the customer together with a membership card. |
| **Alternative Course(s):** | **Step 10:** a) If customer service specialist has not clicked confirm button 30 minutes after the registration id was generated and printed out. Member info is deleted from the temporary registrations list. Member must do the registration process again from the start.  **Step 12:** If the member account cannot be stored in the database display an error window with text  “Database error, please contact the I.T. specialist”. | |
| **Precondition:** | None. | |
| **Postcondition:** | The membership card, membership number and password are given to the member. | |
| **Assumptions:** | None. | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Iput Check. This is a <<Include>> use case for Register use case | |
| **Actor(s):** | Member | |
| **Description:** | This use case describes the membership registration process | |
| **Reference:** | TS-1 | |
| **Typical Course of Events:** | **Actor Action** | **System Response** |
| **Step 1**: Initiate this use case when system has received input information from registration form | **Step 2**: First check whether all the fields are filled,  **Step 3:** Checking that the email, mobile number and date are in the correct format and  **Step 4:** Checkthat there is no user with the same email address in the database.  Conclude this use cases, when all the checks have returned a positive result. |
| **Alternative Course(s):** | **Step 2** If all the fields are not filled: put text “This field is compulsory” next to the window and don’t continue to next step.  **Step 3** If email or mobile is in wrong format put “Email is not correct” next to email textbox. “Number is not Hong Kong format” next to mobile textbox. “Date is not DD/MM/YYYY” format.  **Step 4** If there is an already existing email in the database display error window with text “User account already in the database, go to the Customer Account Specialist” to get more information. Don’t continue to next step. | |
| **Precondition:** | None. | |
| **Postcondition:** | The membership card, membership number and password are given to the member. | |
| **Assumptions:** | None. | |

## 2. Model object interaction and behaviors for the Design Use Case “Register”.

### i. Identify and categorize design classes;

|  |  |  |
| --- | --- | --- |
| Boundary | Control | Entity |
| Customer registration Main window  Customer Service Main window  Registration Form window  Notification window  Printer interface.  Current registrations window  Confirmation window  Successful account creation window  Error window | Registration controller  Input Check controller  Printer controller | Member  Purchase  Monthly Award |

### ii. Identify (additional) attributes;

email

birthday

password

password repeat

mobile phone number

email text

cur\_member

store\_time

window specific texts stored as a string

### iii. Identify behaviors and responsibilities;

|  |  |  |
| --- | --- | --- |
| Behaviour | Automatic/Manual | Class Type |
| clicks on the button register | MANUAL |  |
| Display a form window, | AUTOMATIC | BOUNDARY |
| Input the personal information | MANUAL |  |
| click submit. | MANUAL |  |
| check whether all the fields are filled | AUTOMATIC | BOUNDARY |
| Check that email, mobile number and date are in the correct format | AUTOMATIC | BOUNDARY |
| Check that there is no user with the same email address in the database. | AUTOMATIC | CONTROLLER |
| Display a notification window | AUTOMATIC | BOUNDARY |
| Generate randomly registration ID | AUTOMATIC | CONTROLLER |
| Print out a randomly generated registration id. | AUTOMATIC- | CONTROLLER |
| Display main screen | AUTOMATIC | BOUNDARY |
| Store the data together with the Registration ID for 30 minutes to the temporary registrations list | AUTOMATIC | C |
| Pick the printed paper with the registration key up | Manual |  |
| receives the slip and inputs it in the registration | Manual |  |
| DisplayConfirmation window | T | BOUNDARY |
| Customer service specialist clicks confirm. | MANUAL |  |
| Generate membership number in format TSENTITY(current amount of customers)+1. | AUTOMATIC | CONTROLLER |
| Store the registration info together with the membership number | AUTOMATIC | ENTITY |
| Send a message to the card printer toAssign the card to the membership ID and print out the plastic card | AUTOMATIC | BOUNDARY |
| Send Entitymail to the customer | AUTOMATIC | CONTROLLER |
| Receive the printed membership card |  |  |
| Go to the card pickup counter | MANUAL |  |
| Display successful account creation window | AUTOMATIC | BOUNDARY |
| Receive Card | MANUAL |  |
| display an error window with text  “Database error, | AUTOMATIC | BOUNDARY |
| : put text “This field is compulsory” next to the relevant textboxes | AUTOMATIC | BOUNDARY |
| put “Email is not correct” next to email textbox. “Number is not Hong Kong format” next to mobile textbox. “Date is not DD/MM/YYYY” format next tot DateBox. | AUTOMATIC | BOUNDARY |
| Display error window with text “User account already in the database, go to the Customer Account Specialist” | AUTOMATIC | BOUNDARY |
| . Member info is deleted from the temporary registrations list. | AUTOMATIC | CONTROLLER |

**Function mapping to classes.**

|  |  |  |  |
| --- | --- | --- | --- |
| Behaviour | Class Type | Class representation | Class |
| Display a form window, | BOUNDARY | Display(Form\_window) | Customer registration Main window : Inherited from Abstract class Main window |
| Check whether all the fields are filled | BOUNDARY | InputCheck(Form) | Registration Form window |
| proceed checking that the email, mobile number and are in the correct format | BOUNDARY | InputCheck(Form) | Registration Form window |
| Check that there is no user with the same email address in the database. | CONTROLLER | ifExists(Member ) | InputCheck controller |
| Display a notification window | BOUNDARY | Display(Notification\_window) | Customer registration Main window : Inherited from Abstract class Main window |
| Generate randomly registration ID | CONTROLLER | GenerateRegID() | Registration controller |
| Print out a randomly generated registration id. | CONTROLLER | Print(Text) | Printing interface |
| Display main screen | BOUNDARY | Display() | Customer registration Main window : Inherited from Abstract class Main window |
| Store the data together with the Registration ID for 30 minutes to the temporary registrations list | C | AddToTempList(registration\_id, Member,store\_time) | Registration controller |
| Confirmation window | BOUNDARY | Display(Confirmation window) | Customer Service Main window  : Inherited from Abstract class Main window |
| Generate membership number in format TSENTITY(current amount of customers)+1. | CONTROLLER | GenerateID | Registration controller |
| store the registration info together with the membership number | E | Store(Member) | Member |
| Send a message to the card printer toAssign the card to the membership ID and print out the plastic card | C | PrintMemberCard(Member\_Id) | Printer Interface |
| Send Entitymail to the customer | CONTROLLER | SendEmail(Member\_email) | Registration controller |
| Customer Service Main window | CONTROLLER | Display(Success) | Customer Service Main window |
| display an error window with text: “Database error” | BOUNDARY | DisplayError (Database) | Customer Service Main window  : Inherited from Abstract class Main window |
| Put text “This field is compulsory” next to the window | BOUNDARY | CompulsoryField() | Customer registration Main window : Inherited from Abstract class Main window |
| Put “Email is not correct” next to email textbox. “Number is not Hong Kong format” next to mobile textbox. “Date is not DD/MM/YYYY” format | BOUNDARY | Formating() | Customer registration Main window : Inherited from Abstract class Main window |
| display error window with text “User account already in the database, go to the Customer Account Specialist” | BOUNDARY | DisplayError(errortext) | Customer registration Main window : Inherited from Abstract class Main window |
| Member info is deleted from the temporary registrations list. | CONTROLLER | DeleteRegInfo(Registration\_Id); | Registration Controller |

**Additional functions which**

|  |  |  |
| --- | --- | --- |
| InitiateRegistration(Input) | CONTROLLER | Registration  controller |
| CreateMemberObject(Input) | CONTROLLER | Registration controller |
| RetrieveFromList(Registration\_Id) | CONTROLLER | Registration controller |
| InitiateRegistration(Input) | CONTROLLER | Registration  controller |
| Timechecker()  *Checks that unfinished registrations are not stored in the list for a longer time than 30. minutes.* | CONTROLLER | Registration  controller |

**Function call playthrough to clarify behaviour flow**

Registration Form window.InputCheck(Form)->Registration\_controller.InitiateRegistration(input) -> Registration\_controller.CreateMemberObject(Input) -> Input\_controller.ifExists(Member)->Registration controller. GenerateRegID()->Registration\_controller.AddToTempList(registration\_id, Member,store\_time)->Printer interface.Print(registration\_id)-> Customer registration Main window.Display(Notification window).->Customer Registration Main window.Display()

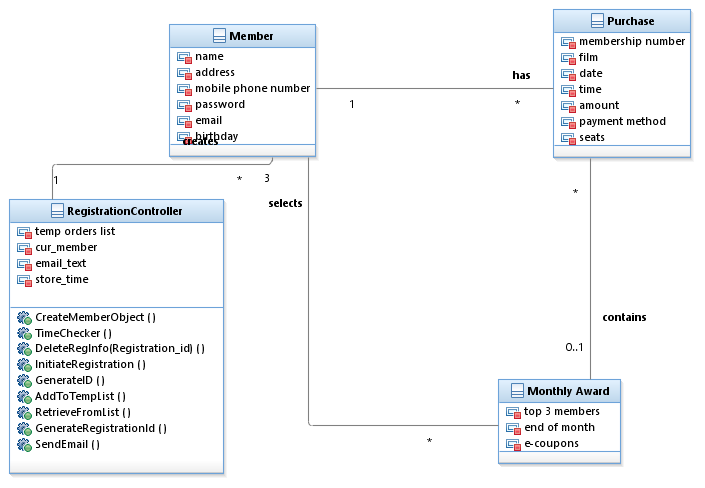
Registration Controller. RetrieveFromList(Registration\_ID)->Registration controller.GenerateID()-> Member.Store(Member)->Printer Interface.PrintMemberCard(Member\_Id)->Registration controller.sendEmail(email)-> Customer Service Main window.Display(Success)

## 3. Communication diagram(s) for the Design Use Case “Register”.

****

## 4. Update the class diagram from the one appeared in Assignment 1 sample solution so that it *only* reflects (i) the changes of object classes relevant to the Design Use Case “Register”, and (ii) design class identified above in Step 2.

Create, Delete, Methods, date methods for all entity classes are assumed.

****

# Part II: After the Requirement Change

The project management team and the executives have reviewed the requirements for OO Analysis. With the consideration of emerging mobile technology, no membership card will be issued after registration. They would like you to send the electronic membership card by email to the members. Therefore, you are asked to rEntitywork the above documentation.

Repeat the above Steps 1 to 4 to reflect the new requirements. The deliverables are design use case description, list of revised design classes with behaviors and responsibilities, an updated class diagram (based on Design Use Case “Register”), and communication diagrams in sketch.

Note: You do not need to redo everything for part 2 of the assignment, but only highlight their changes.

## 1. Transform the analysis use case documentation of use case “Register” to design use case documentation.

Use case stays the same, except for Step 13, Step 14, 16, Conclusion text in use case Registe.

Step 13 is replaced by

**Step 13:** Send a message to the printer interface to generate an electronic member card for the current member-id

Step 14 is modified in the following way, where the part colored by xxxx displays added text:

**Step 14:** Add the electronic id card as an attachement. Send Entitymail to the customer with text: Dear Mr/Mrs <Name> Your account has been created. Your membership number is <Membership number> and your password is <password>. We have added the electornic id card as an attachement to this email.

**Step 16:** Give confirmation to the member that the member account has been created.

Conclusion text changes as follows: Conclude the use case when a membership number and a password is assigned to the customer

## 2. Model object interaction and behaviors for the Design Use Case “Register”.

### i. Identify and categorize design classes;

Nothing changes

### ii. Identify (additional) attributes;

Nothing changes.

### iii. Identify behaviors and responsibilities;

|  |  |  |  |
| --- | --- | --- | --- |
| Send a message to the card printer toAssign the card to the membership ID and print out the plastic card | BOUNDARY | PrintMemberCard(Member\_Id) | Printer Interface |

Changes to

|  |  |  |  |
| --- | --- | --- | --- |
| Send a message to the printer interface to generate an electronic member card for the current member-id | BOUNDARY | PrintElectronicMemberCard(Member\_Id) | Printer Interface |

## 3. Communication diagram(s) for the Design Use Case “Register”.

## (To ease student workload, low-level interactions with sequence diagram(s) and state machine(s) need not be prepared for this assignment.)

****

## 4. Update the class diagram from the one appeared in Assignment 1 sample solution so that it *only* reflects (i) the changes of object classes relevant to the Design Use Case “Register”, and (ii) design class identified above in Step 2.

Nothing changes