**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Describe the Run-time Architecture**

**for**

**The Automatic**

**Attendance Checking System**

**Version 1.1**

**Prepared by Huynh Vinh Nam**

**Le Huy Duc**

**Cao Phuong Linh**

**OOAD Group 2**

**17-Dec-2018**

**Table of Contents**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for Changes** | **Version** |
| Huynh Vinh Nam | 17-Dec-2018 | Create document template | 1.0 |
| Huynh Vinh Nam | 21-Dec-2018 | Add Picture from model file | 1.1 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**1. Introduction**

**1.1. Purpose**

This is a report on the subject Object-oriented Analysis and Design of group two, class ICT-BI7 about Run-time Architecture Solution.

The report is written based on the reporting format “IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications”. Content-based report is accepted and be satisfied with group meeting minute on 18-Dec-2018 (Viber teleconference).

**1.2. Intended Audience and Reading Suggestions**

*The different types of reader that the document is intended for are:*

**● Project managers:** who manage and take respond for the quality of the system. Project

managers should read the whole document for planning and assigning work.

**● Developers:** Dev is the person who implement the system from the design and documents into a runnable version. Dev have to read the whole document to implement the right system.

**● Documentation writers:** who will write the future document (report, minutes).

Documentation writers should read to understand the Use Case Main Diagram part.

This section describes the system’s decomposition into lightweight processes (single threads of control) and heavyweight processes (groupings of lightweight processes). Organize the section by groups of processes that communicate or interact. Describe the main modes of communication between processes, such as message passing, interrupts, and rendezvous.

**1.3. Product Scope**

The software’s main users are students and lecturers. Software will create an environment where user (student) can check for the attendance and user (lecturer) can view and/or manage the attendance list in the course(s).

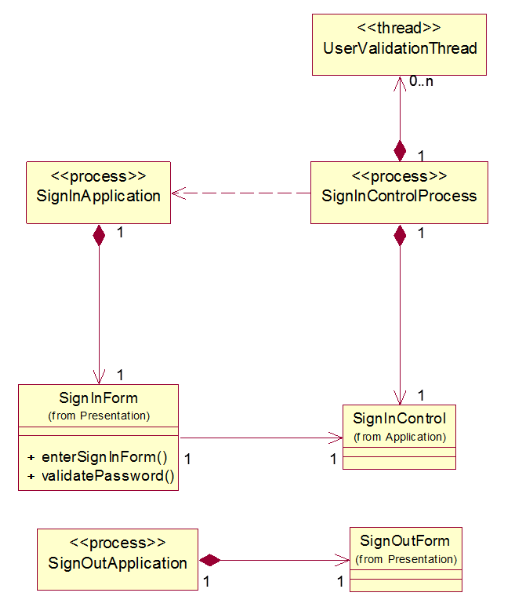
**1.4. References**

[1] Form of presentation IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

**2. Describe Partly**

**2.1. In and Out**

**2.1.1 Process model**

****

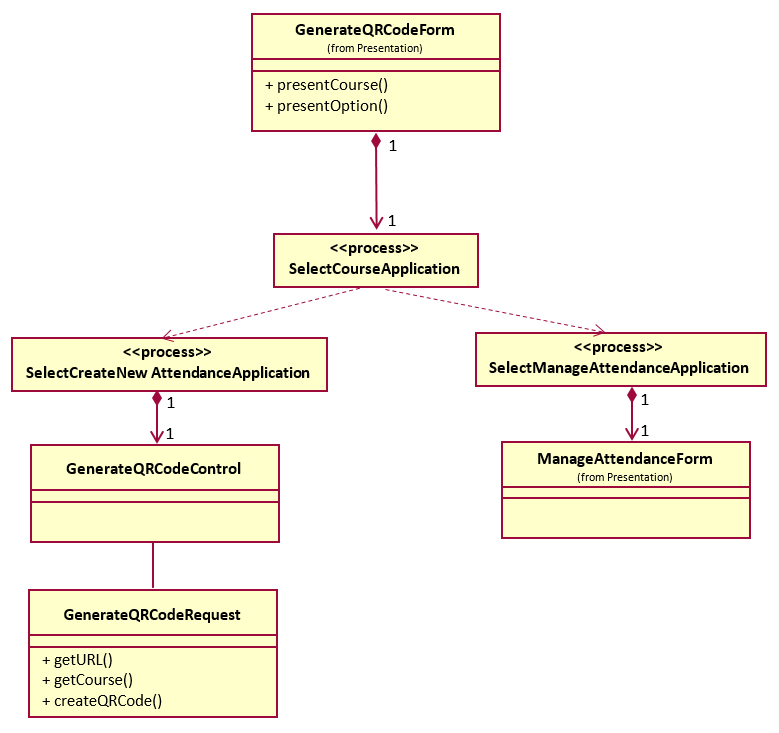
*Figure 1: In and Out Process model*

**2.1.2 Process Element Descriptions**

* **SignInApplication:** Controls the interface of the Sign In application. Controls the family of forms that the users use to sign in. There is one instance of this process.
* **SignInControl Process:** Manages execution of the sign in processing after users submit their username and password to log into the system. There is one instance and one thread of this process each time the username and password of user is validated.
* **SignOutApplication:** Controls the interface of the Sign Out application. Controls the family of forms that the users use to sign out. There is one instance of this process.

**2.2. Generate QR Code**

**2.2.1 Process model**

****

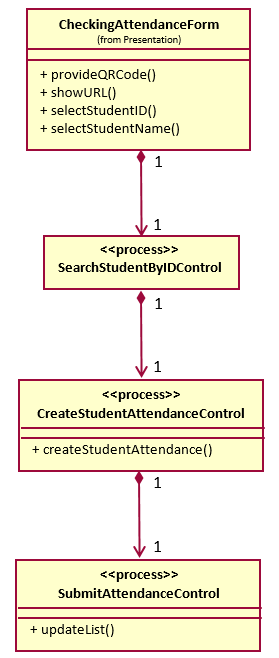
*Figure 2: Generate QR Code Process model*

**2.2.2 Process Element Descriptions**

* **SelectCourseApplication:** Controls the interface of the Select Course application. Controls the two options that the users may use to Create New Attendance or Manage Attendance. There are two different threads of this process.
* **SelectCreateNewAttendance Process:** Manages execution of the selected course processing after users confirms the action to create a new attendance. There is one instance for each time the user’s create a new attendance action is verified.
* **SelectManageAttendance Process:** Manages execution of the selected course processing after users confirms the action to manage existing attendance(s). There is one instance for each time the user chooses to manage attendance is confirmed.

**2.3. Checking Attendance**

**2.3.1 Process model**

****

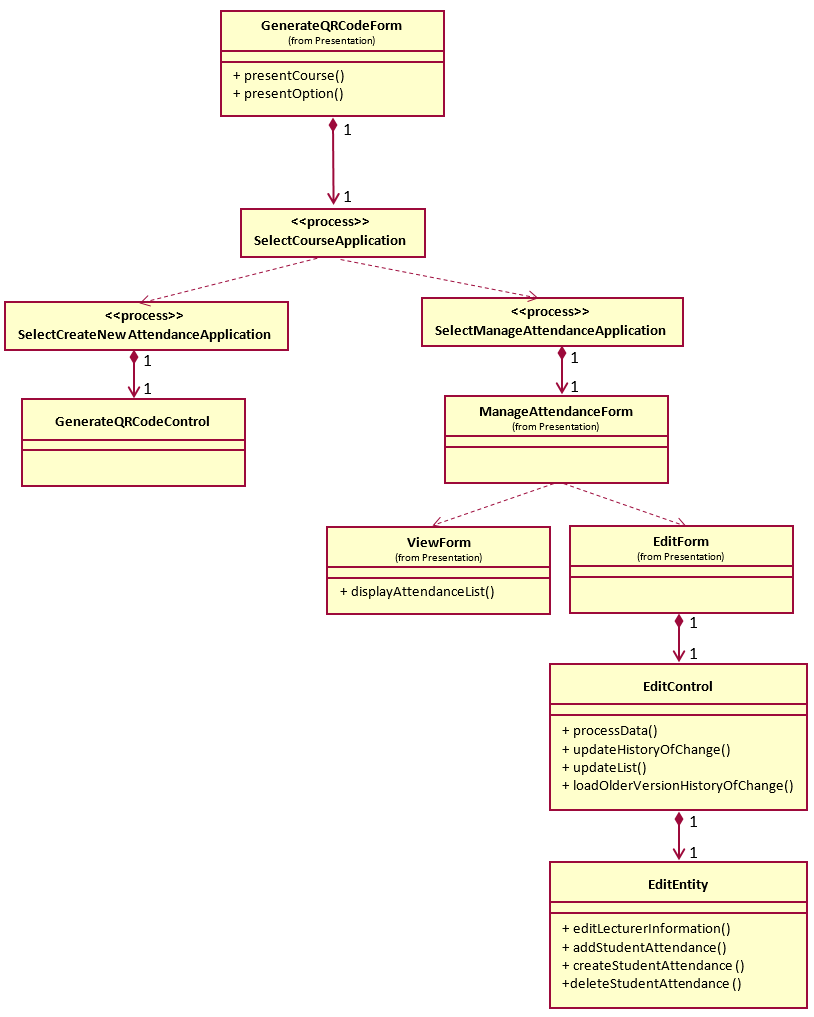
*Figure 3: Checking Attendance Process model*

**2.3.2 Process Element Descriptions**

* **CheckingAttendanceForm:** Controls the interface of the Checking Attendace application. There is one instance of this process.
* **SearchStudentByIDControl Process:** Manages execution of the Student’s ID searching action. There is one instance for this process.
* **CreateStudentAttendanceControl Process:** Manages execution of the create a new attendance for the found student. There is one instance for this process.
* **SubmitAttendanceControl Process:** Update the attendance list after a successful attendance created for the current student.

**2.4. Manage Attendance list**

**2.4.1 Process model**

****

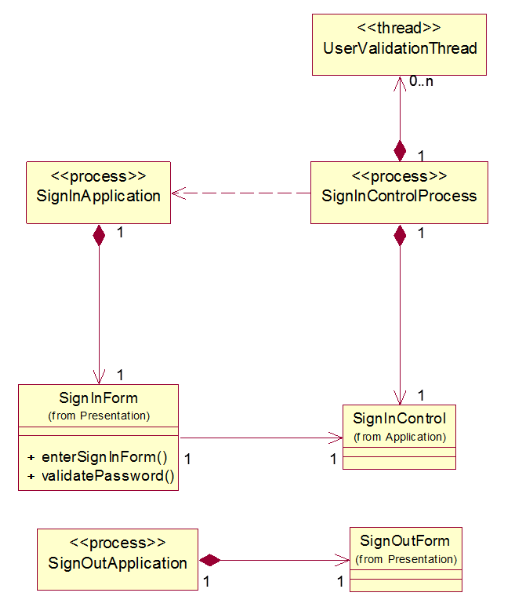
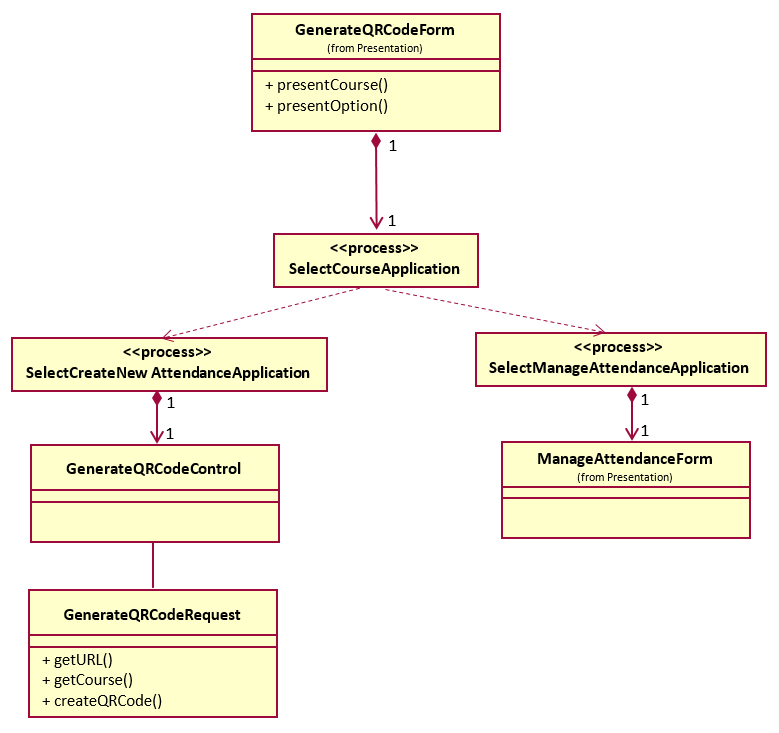
*Figure 4: Manage Attendance list Process model*

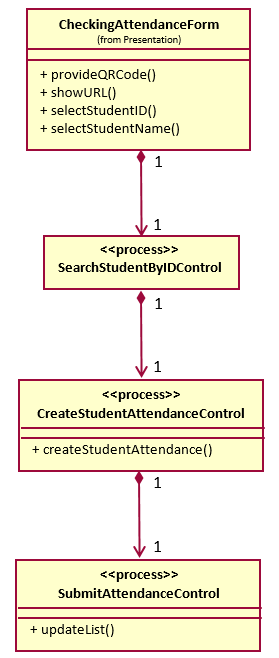
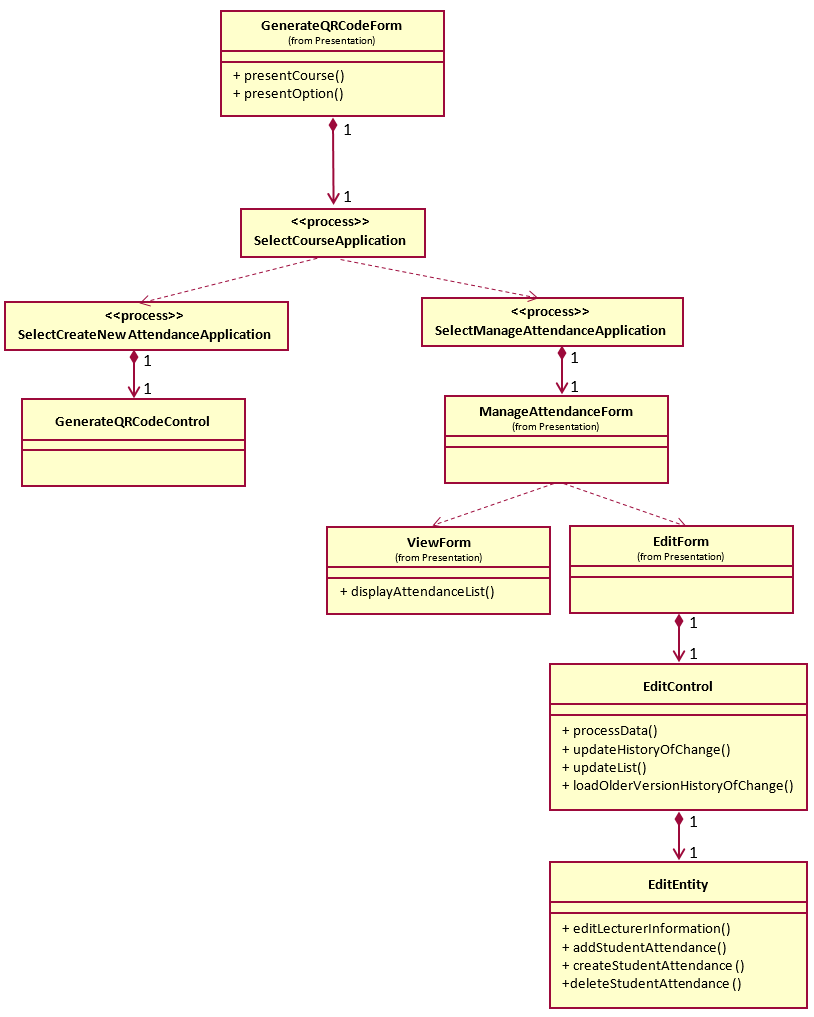
**2.4.2 Process Element Descriptions**

* **SelectCourseApplication:** Controls the interface of the Select Course application. Controls the two options that the users may use to Create New Attendance or Manage Attendance. There are two different threads of this process.
* **SelectManageAttendance Process:** Manages execution of the selected course processing after users confirms the action to manage existing attendance(s). There is one instance for each time the user’s create a new attendance action is verified.
* **ManageAttendanceForm:** Controls the interface of the Manage Attendace application. There are two different threads of this process.
* **ViewForm:** Controls the interface of the View choice.
* **EditForm:** Controls the interface of the Edit choice. There is one instance of this process.
* **EditControl Process:** Manages execution to make changes to the existing attendance. There is one instance for this process.

**3. Describe Concurrency**

**3.1. Process model**

** **

** **

*Figure 5: Concurrency Process model*

**3.2. Process Element Descriptions**

* **SignInApplication:** Controls the interface of the Sign In application. Controls the family of forms that the users use to sign in.
* **SignInControl Process:** Manages execution of the sign in processing after users submit their username and password to loggin to the system.
* **SignInApplication:** Controls the interface of the Sign Out application. Controls the family of forms that the users use to sign out.
* **SelectCreateNewAttendance Process:** Manages execution of the selected course processing after users confirms the action to create a new attendance. There is one instance for each time the user’s create a new attendance action is verified.
* **SelectManageAttendance Process:** Manages execution of the selected course processing after users confirms the action to manage existing attendance(s). There is one instance for each time the user chooses to manage attendance is confirmed.
* **SearchStudentByIDControl Process:** Manages execution of the Student’s ID searching action. There is one instance for this process.
* **CreateStudentAttendanceControl Process:** Manages execution of the create a new attendance for the found student. There is one instance for this process.
* **SubmitAttendanceControl Process:** Update the attendance list after a successful attendance created for the current student.
* **SelectCourseApplication:** Controls the interface of the Select Course application. Controls the two options that the users may use to Create New Attendance or Manage Attendance. There are two different threads of this process.
* **SelectManageAttendance Process:** Manages execution of the selected course processing after users confirms the action to manage existing attendance(s). There is one instance for each time the user’s create a new attendance action is verified.
* **EditControl Process:** Manages execution to make changes to the existing attendance. There is one instance for this process.