< 분산 프로그래밍 >

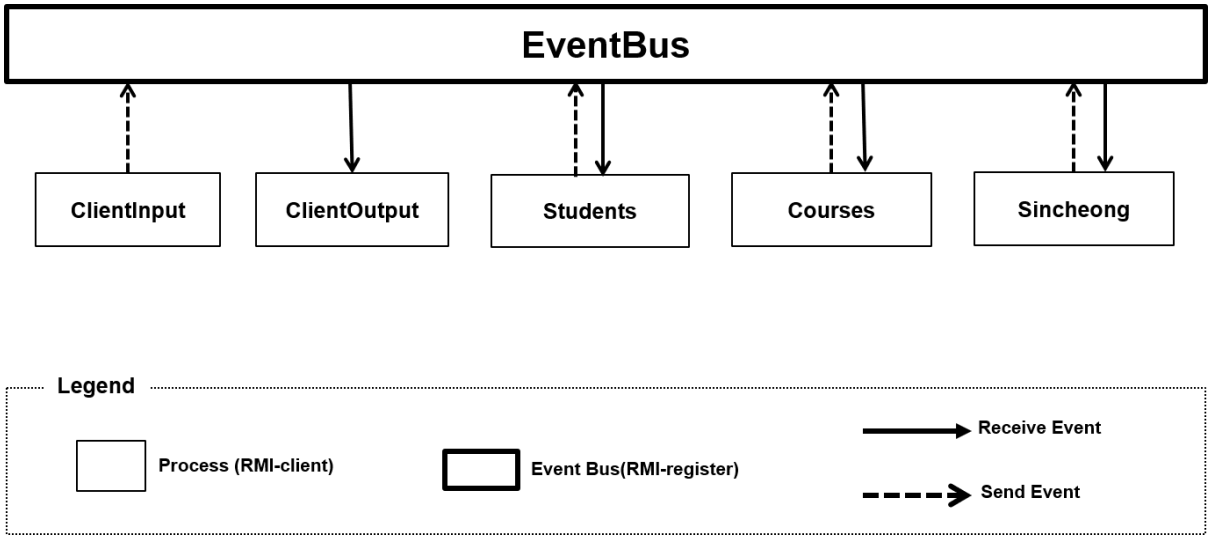
중간 REPORT

목차

과목명	분산프로그래밍	
교수명	김정호	
학과	융합소프트웨어학부	
학번	60161606	60161611
이름	박준현	성소연

1. 설계

1) Component 설계

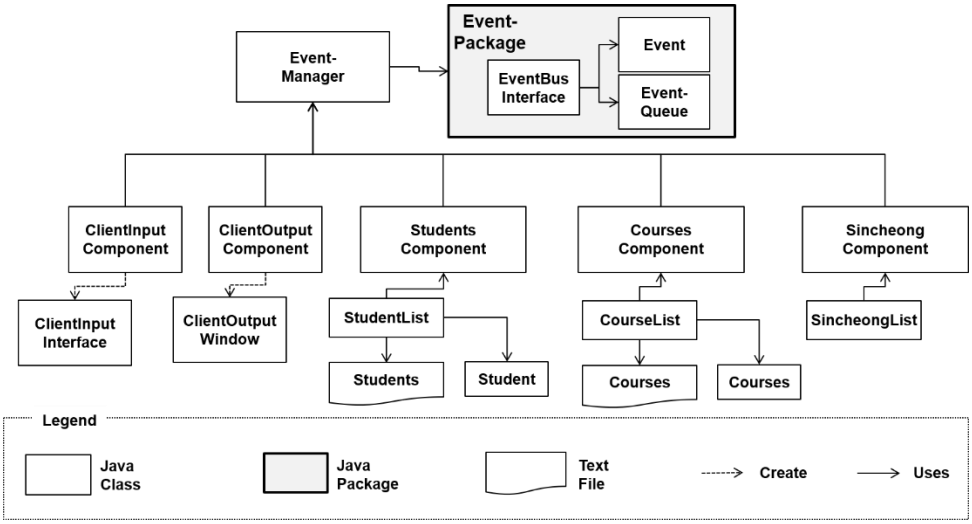


- 각 컴포넌트들과 EventManager는 RMI를 통해 통신을 하게 된다.
- 각 컴포넌트들은 Interface내부에 있는 Naming.lookup("EventBus"); 함수를 통해서 EventBus에 연결하게 된다.
- 컴포넌트가 등록되었는지 registered로 판별하고 true이면 컴포넌트가 실행된다.
- 컴포넌트들은 각각 별개로 실행되고 직접적인 연관이 없으며 우선실행된 EventBus와 연결을 맺는다. 한 컴포넌트를 종료하더라도 다른 컴포넌트들은 무관하게 실행이 지속된다.
- 각 컴포넌트들은 While문을 통해서 EventBus를 polling 하다가 자신이 처리해야 할 EventId가 있으면 처리한다.
- EventBus에 Event를 보낼 때는 Interface에 정의되어 있는 sendEvent함수를 통해 EventId와 String 값으로 메시지를 전달한다.

<컴포넌트별 역할>

- ClientInputComponent : 시스템 사용자가 입력값을 줄 수 있는 Text-based User Interface 컴포넌트로 사용자에게 받은 Input값들을 EventBus로 보낸다.
- ClientOutputComponent : 사용자가 요청한 기능들의 결과를 확인할 수 있는 컴포넌트로 모든 기능의 결과값을 출력하여 User에게 보여준다.
- StudentsComponent : Student 정보와 관련된 기능을 처리하는 컴포넌트이다. 학생리스트 조회, 학생등록, 학생삭제, 학생판별의 역할을 한다.
- CoursesComponent : Course 정보와 관련된 기능을 처리하는 컴포넌트이다. 수업리스트조회, 수업등록, 수업삭제, 수업판별, 선수과목판별의 역할을 한다.
- SincheongComponent : Student가 Course를 신청하는 기능을 처리하는 컴포넌트이다. (학생의)수업신청, 수업리스트조회의 역할을 한다.

2) Class 설계



- 큰 구성은 EventPackage, EventManager, Components 들로 이루어진다.
- EventPackage는 Event를 정의하고 관리하는 역할을 한다. EventBusInterface는 RMI 서버인 EventBus와 통신하기 위한 인터페이스 역할을 하는 클래스로 register, unregister 메소드와 Event전송 메소드인 sendEvent와 EventQueue를 반환하는 메소드가 있다 하지만 이 클래스가 실제 인터페이스는 아니다. 컴포넌트들이 주고받는 EventId는 enum type으로 정의되어 있다. EventQueue에서는 Event들을 저장하는 역할을 한다.

- EventBus라는 이름으로 EventManager를 구현하였다. RMI 메세지 서버 클래스로 각 컴포넌트들에 대한 EventQueue 객체를 갖고 있고 Event 전송 요청이 왔을 때 해당 Event를 모든 EventQueue 객체에 추가하여준다. 발생한 Event들을 가지고 있는 클래스이다.

- Component들은 ClientInput, ClientOutput, Students, Courses, Sincheong 컴포넌트가 있다.

- ClientInputComponent 는 사용자들의 Input을 받아서 Event로 보내는 역할을 한다. 오직 이벤트를 보내기만 하고 받는 기능은 없다.
  ClientOutputComponent 는 컴포넌트들에서 나온 결과값들을 출력하는 역할을 한다. 오직 결과값들을 출력하는 역할을 하고 이벤트를 보내는 기능은 없다.

- StudentsComponent, CoursesComponent 에는 각각 student.txt 와 courses.txt파일을 가지고 있다. 이 txt 파일들을 StudentsList 와 CoursesList가 읽어서 어레이에 저장하게 된다. Student 와 Course 클래스는 1개의 student 혹은 course 의 속성들을 저장하기 위해 만들어진 클래스인데, 들어오는 inputString을 끊어서 속성을 저장하는 역할을 한다.
  이 어레이를 통해서 StudentsComponent 와 CoursesComponent는 리스트 출력, 신규 학생과 수업 등록, 기존 학생과 수업 삭제, 학생과 수업의 존재 판별과 수업의 선수과목을 판별한다.

- SincheongComponent 는 수강신청을 하는 학생 중 판별되어 넘어온 학생과 수업의 값을 저장하는 역할과 신청리스트를 출력하는 역할을 한다. SincheongList에서 신청정보를 저장하고 있다.

2. 코드

1) 코드리스트

- ClientInputComponent : 7. RegisterSincheong , 8.ListSincheong

- StudentsComponent : EventId.DeleteStudents , EventId.CheckIfStudentExist

- CoursesComponent : EventId.DeleteCourses , EventId.CheckIfCourseExist

- SincheongComponent : EventId.RegisterSincheong , EventId.ListSincheong

- SincheongList : ArrayList vSincheong

2) 코드코멘트

: 코드 코멘트는 주석으로 표시하였습니다.

StudentsComponent

- DeleteStudents

StudentsComponent.java

```
75
76      } else if (event.getEventId() == EventId.DeleteStudents) {
77          // Student를 삭제한다.
78          String ID = event.getMessage();
79          if (studentsList.isRemovedStudent(ID) == false) {
80              // 만약 이미 존재하는 Student라면 vStudent 어레이에서 해당 ID의 학생을 삭제한다.
81              studentsList.vStudent.remove(ID);
82              System.out.println("A student is successfully removed...");
83              System.out.println("\"" + ID + "\"");
84          } else {
85              // 만약 존재하지 않는 Student라면 이미 지워진 학생이라는 메세지를 띄운다.
86              System.out.println("\n ** Sending an event(ID:" + EventId.ClientOutput + ") with");
87              System.out.println("\n ** Message: This student is already removed.  ...");
88              eventBusInterface.sendEvent(new Event(EventId.ClientOutput, "This student is already removed."));
89          }
90
```

- CheckIfStudentExsit

StudentsComponent.java

```
91      } else if (event.getEventId() == EventId.CheckIfStudentExist) {
92          // Student가 어레이에 존재하는지를 체크한다.
93          String check = event.getMessage();
94          if (studentsList.isRegisteredStudent(check) == false) {
95              // 존재하지 않는 학생일 때 등록되지 않았다고 메세지를 띄운다.
96              System.out.println("등록되지 않은 학생입니다.");
97              eventBusInterface.sendEvent(new Event(EventId.ClientOutput, "등록되지 않은 학생입니다."));
98              System.out.println("\n ** Sending an event(ID:" + EventId.ClientOutput + ") with");
99              System.out.println("\n ** Message: 등록되지 않은 학생입니다.  ...");
100
101         } else {
102             // 존재하는 학생일 때 등록된 학생이라는 메세지를 띄우고 RegisterSincheong 이벤트를 보낸다.
103             System.out.println("등록된 학생입니다.");
104             System.out.println("\n ** Sending an event(ID:" + EventId.ClientOutput + ") with");
105             System.out.println("\n ** Message: This student does exist  ...");
106             // 신청리스트를 만들어서 신청학생의 ID 값을 넣는다
107             sincheongList = new SincheongList();
108             sincheongList.vSincheong.add(check);
109             System.out.println(sincheongList.vSincheong);
110             sl.StudentID = check;
111             eventBusInterface.sendEvent(new Event(EventId.RegisterSincheong, check));
112         }
```

CoursesComponent

- DeleteCourses

CoursesComponent.java

```
73  }else if(event.getEventId() == EventId.DeleteCourses) {
74      // Course를 삭제한다.
75      String ID = event.getMessage();
76      if(coursesList.isRemovedCourse(ID) == false){
77          // 만약 이미 존재하는 Course라면 vCourse 어레이에서 해당 ID의 학생을 삭제한다.
78          coursesList.vCourse.remove(ID);
79          System.out.println("A course is successfully removed...");
80          System.out.println("\"" + ID + "\"");
81      } else{
82          // 만약 존재하지 않는 Course라면 이미 지워진 수업이라는 메세지를 띄운다.
83          System.out.println("\n ** Sending an event(ID:" + EventId.ClientOutput + ") with");
84          System.out.println("\n ** Message: This course is already removed.  ...");
85          eventBusInterface.sendEvent(new Event(EventId.ClientOutput, "This course is already removed."));
86      }
87
```

- CheckIfCourseExist

CoursesComponent.java

```
88  }   else if(event.getEventId() == EventId.CheckIfCourseExist) {
89      // Course가 어레이에 존재하는지를 체크한다.
90      String courseID = event.getMessage();
91      if(coursesList.isRegisteredCourse(courseID) == false){
92          // 존재하지 않는 수업일 때 등록되지 않았다고 메세지를 띄운다.
93          System.out.println("등록되지 않은 과목입니다");
94          eventBusInterface.sendEvent(new Event(EventId.ClientOutput, "등록되지 않은 과목입니다."));
95          System.out.println("\n ** Sending an event(ID:" + EventId.ClientOutput + ") with");
96          System.out.println("\n ** Message: 등록되지 않은 과목입니다.  ...");
97
98      } else{
99          // 존재하는 수업일 때 등록된 과목이라는 메세지를 띄우고 RegisterSincheong 이벤트를 보낸다.
100         System.out.println("등록된 과목입니다");
101         System.out.println("\n ** Sending an event(ID:" + EventId.ClientOutput + ") with");
102         System.out.println("\n ** Message: This course does exist  ...");
103         // 신청리스트를 만들어서 신청과목의 ID 값을 넣는다
104         sincheongList = new SincheongList();
105         sincheongList.vSincheong.add(courseID);
106         System.out.println(sincheongList.vSincheong);
107         eventBusInterface.sendEvent(new Event(EventId.RegisterSincheong, courseID));
108     }
```

SincheongComponent

- ListSincheong

SincheongList.java | SincheongComponent.java

```
        while (!done) {
            eventQueue = eventBusInterface.getEventQueue();
            for (int i = 0; i < eventQueue.getSize(); i++) {
                event = eventQueue.getEvent();
                System.out.println("Received an event(ID: " + event.getEventId() + ")...");
                if (event.getEventId() == EventId.ListSincheong) {
                    ArrayList returnString = sincheongList.vSincheong;
                       String a = returnString.toString();
                       System.out.println("\n ** Sending an event(ID:" + EventId.ClientOutput + ") with");
                       System.out.println("\n ** Message: \n" + returnString + "\n  ...");
                       eventBusInterface.sendEvent(new Event(EventId.ClientOutput, a));

```

- RegisterSincheong

SincheongList.java | *SincheongComponent.java

```
                } else if (event.getEventId() == EventId.RegisterSincheong) {
                    // 체크 완료된 거 받음
                    String Checked = event.getMessage();
                    System.out.println("Checked...");
                    System.out.println("\"" + Checked + "\"");
                    System.out.println("\n ** Sending an event(ID:" + EventId.ClientOutput + ") with");
                    System.out.println("\n ** Message: " + Checked + " 신청 가능.  ...");

                    // 신청리스트를 만들어서 신청 ID 값을 넣는다
                    sincheongList.vSincheong.add(Checked);
                    String a = sincheongList.vSincheong.toString();
                    System.out.println(sincheongList.vSincheong);

                    eventBusInterface.sendEvent(new Event(EventId.ListCourses, null));
```

ClientInputComponent

ClientInputComponent.java | SincheongComponent.java

```
202             } else if (selection.equals("7")) {
203                 String checkS = "";
204                 String checkC = "";
205                 boolean inputIsDone = false;
206                 while (!inputIsDone) {
207                     // 학생리스트를 띄운다.
208                     System.out.println("!!Check Student Information!!");
209                     clientInputComponent.sendClientInput(EventId.ListStudents, null);
210 
211                     System.out.println("\nEnter student ID and press return (Ex. 20131234)>> ");
212                     // 학생 ID를 입력 받는다.
213                     String sSID = br.readLine().trim();
214                     checkS = sSID;
215 
216                     while (true) {
217                         System.out.println("\nIs it correct student information? (Y/N)");
218                         System.out.println(checkS);
219                         String ans = br.readLine().trim();
220                         if (ans.equalsIgnoreCase("y")) {
221                             inputIsDone = true;
222                             break;
223                         } else if (ans.equalsIgnoreCase("n")) {
224                             System.out.println("\nType again...");
225                             checkS = "";
226                             break;
227                         } else {
228                             System.out.println("\nTyped wrong answer");
229                         }
230                     }
231                     System.out.println("\nSending an event(ID:" + EventId.CheckIfStudentExist + ") with");
232                     System.out.println("\n ** Message \"" + checkS + "\" ...");
233                     // StudentsComponent에 학생ID를 판별하라고 CheckIfStudentExist 이벤트 보냄
234                     clientInputComponent.sendClientInput(EventId.CheckIfStudentExist, checkS);
235                     StudentsList studentsList = new StudentsList("Students.txt");
236 
237                     // 학생 ID판별
238                     if (studentsList.isRegisteredStudent(checkS) == false) {
```

ClientInputComponent.java | SincheongComponent.java

```
239                     // 존재하지 않는 학생일 시
240                         System.out.println("존재하지 않는 학생입니다!!! 메인화면으로 돌아갑니다.");
241                         break;
242                     } else {
243                     // 존재하는 학생일 시 과목리스트를 띄운다.
244                         System.out.println("!!Check Course Information!!");
245                         System.out.println("Enter CourseID and press return (Ex.12345)>> ");
246                     }
247 
248                     clientInputComponent.sendClientInput(EventId.ListCourses, null);
249 
250                     // 과목 ID를 입력받는다.
251                     checkC = br.readLine().trim();
252 
253                     while (true) {
254                         System.out.println("\nIs it correct information? (Y/N)");
255                         System.out.println(checkC);
256                         String ans = br.readLine().trim();
257                         if (ans.equalsIgnoreCase("y")) {
258                             inputIsDone = true;
259                             break;
260                         } else if (ans.equalsIgnoreCase("n")) {
261                             System.out.println("\nType again...");
262                             checkC = "";
263                             break;
264                         } else {
265                             System.out.println("\nTyped wrong answer");
266                         }
267                     }
268                     System.out.println("\nSending an event(ID:" + EventId.CheckIfCourseExist + ") with");
269                     System.out.println("\n ** Message \"" + checkC + "\" ...");
270                     // CoursesComponent에 과목ID를 판별하라고 CheckIfCourseExist 이벤트 보냄
271                     clientInputComponent.sendClientInput(EventId.CheckIfCourseExist, checkC);
```

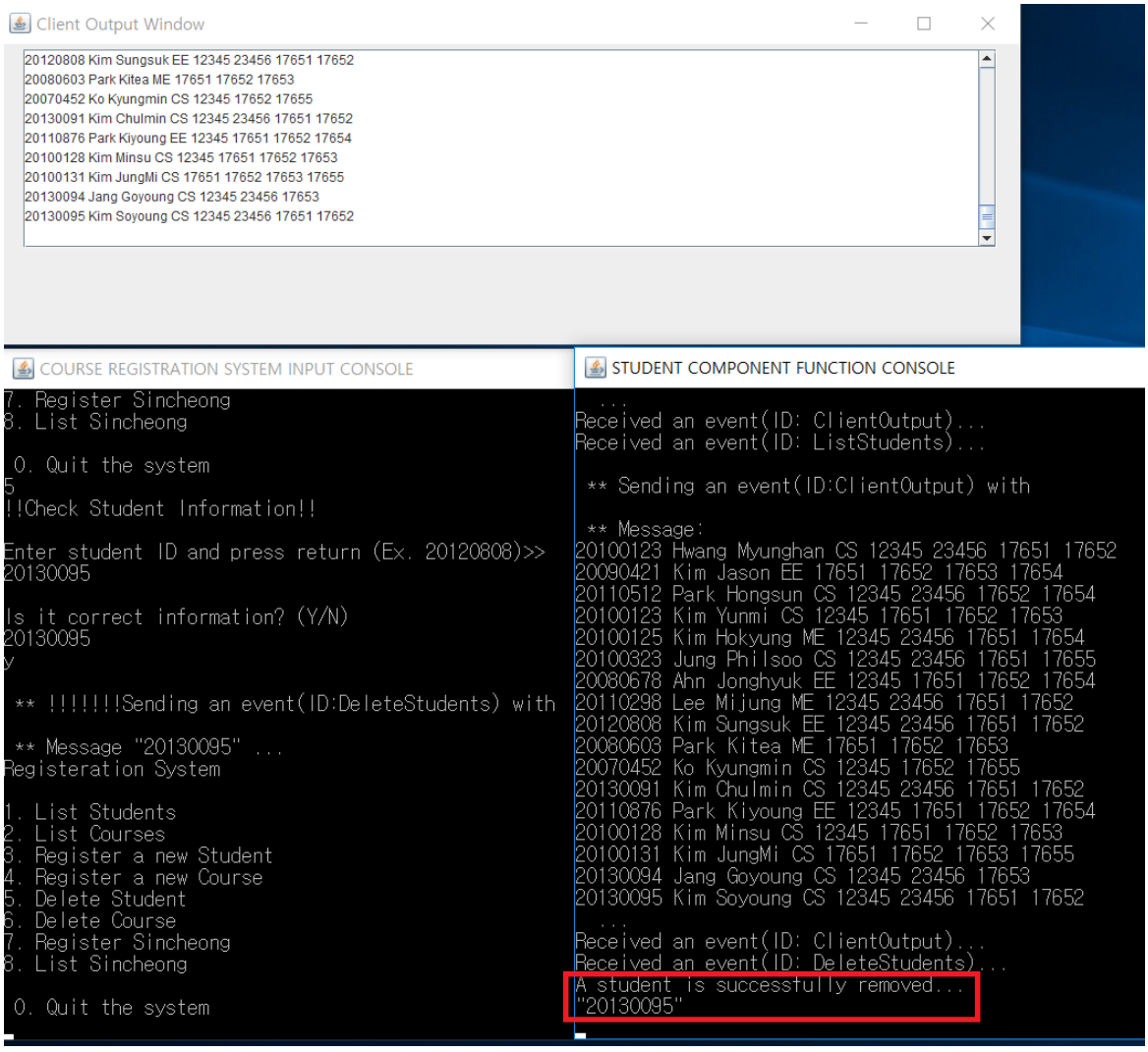
```java
                    CoursesList coursesList = new CoursesList("Courses.txt");

                    // 과목 ID 판별
                    if (coursesList.isRegisteredCourse(checkC) == false) {
                        // 존재하지 않는 과목일 시
                        System.out.println("존재하지 않는 수업입니다!!! 메인화면으로 돌아갑니다.");
                        break;
                    } else {
                        // 존재하는 과목일 시 신청이 완료됨을 띄운다.
                        System.out.println("Sincheong Accessed!!!!!!");
                    }
                }
            } else if (selection.equals("8")) {
                clientInputComponent.sendClientInput(EventId.ListSincheong, null);
```

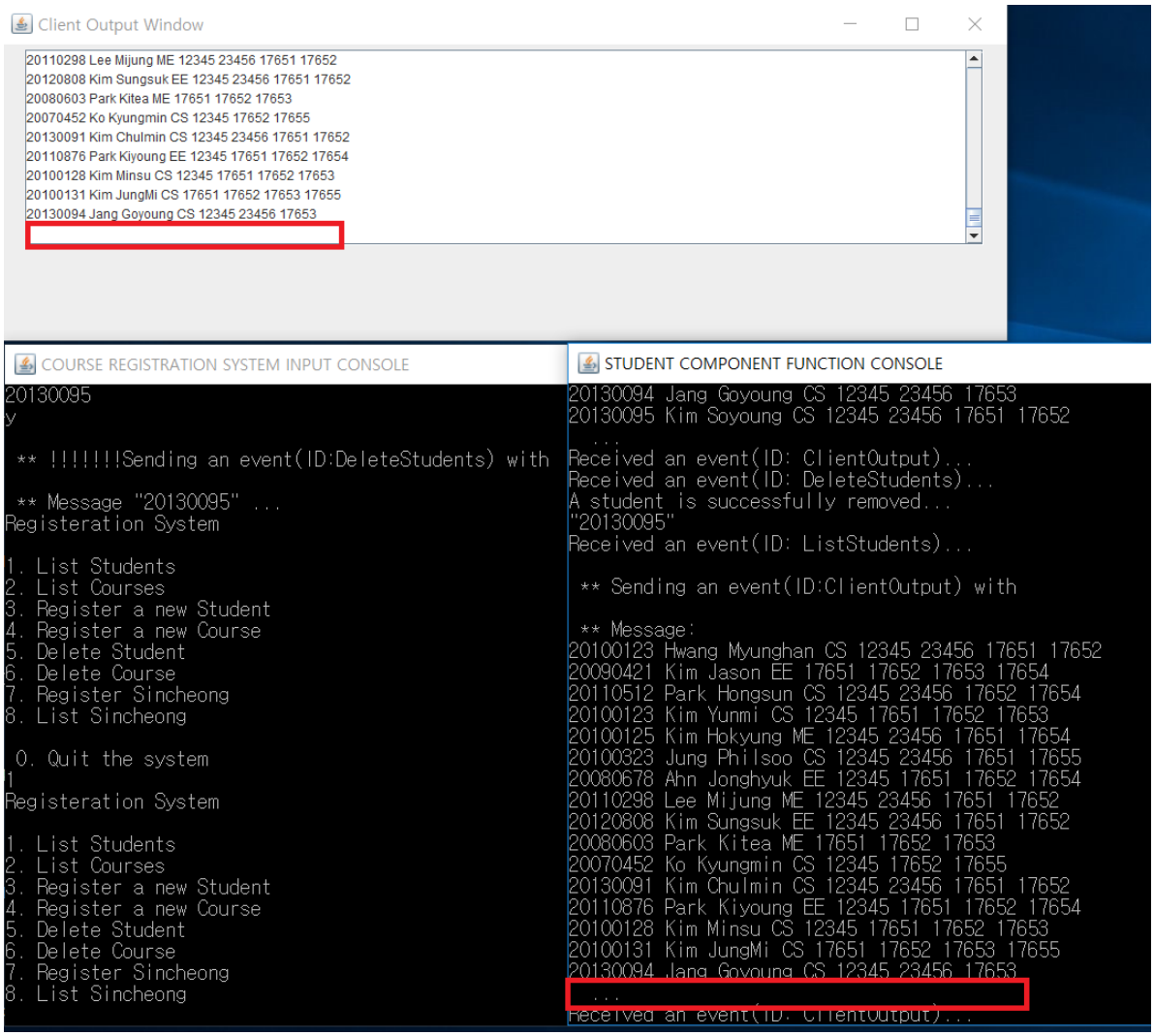
3) 실행결과

DeleteStudents

1. 학생ID입력 후

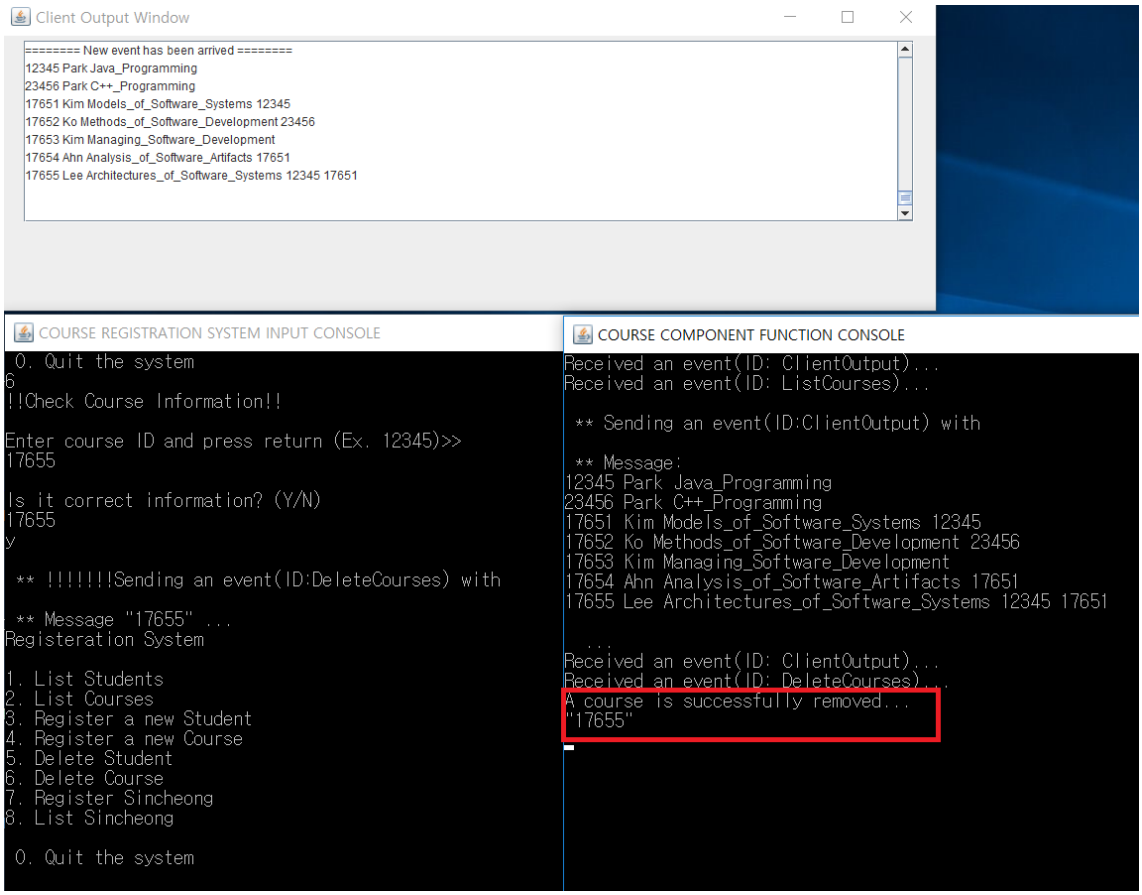


2. 삭제완료 후 학생리스트 출력

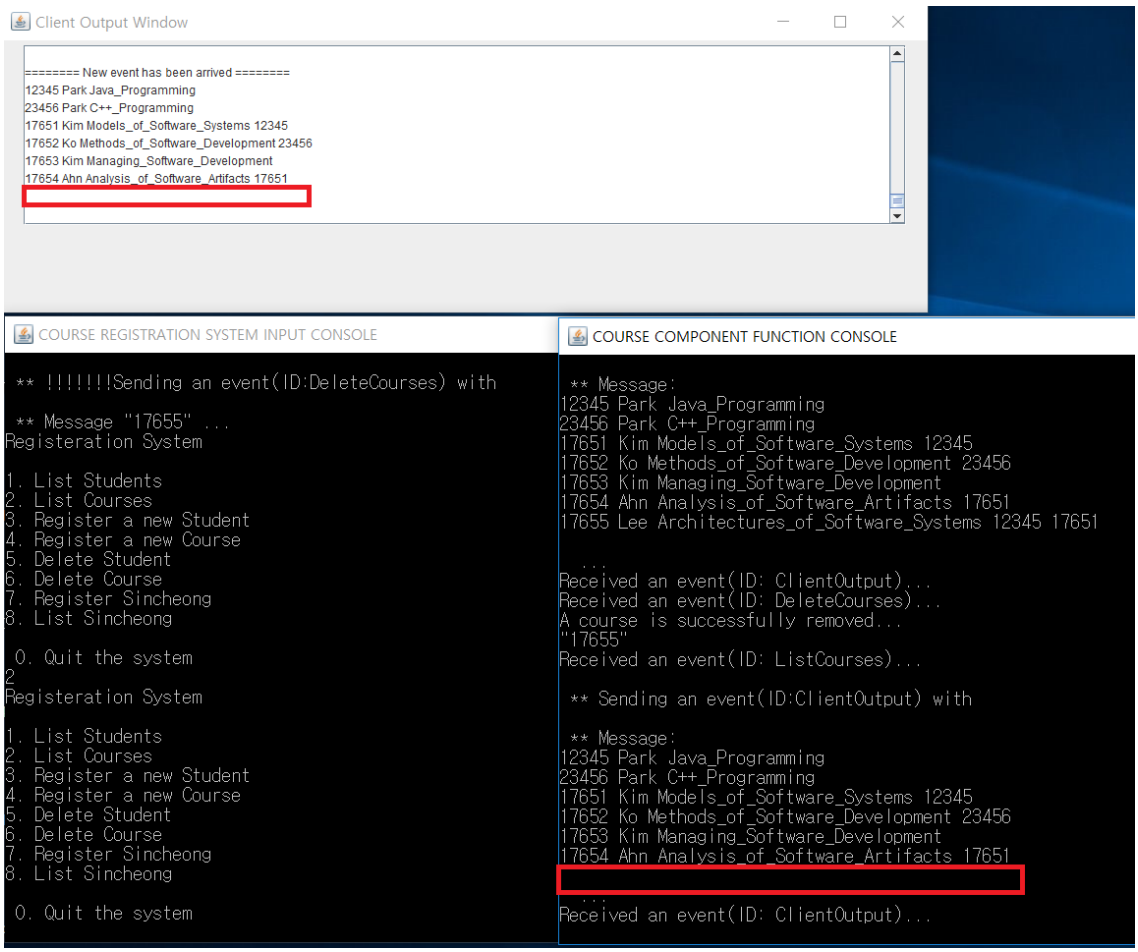


DeleteCourses

1. 과목ID입력 후

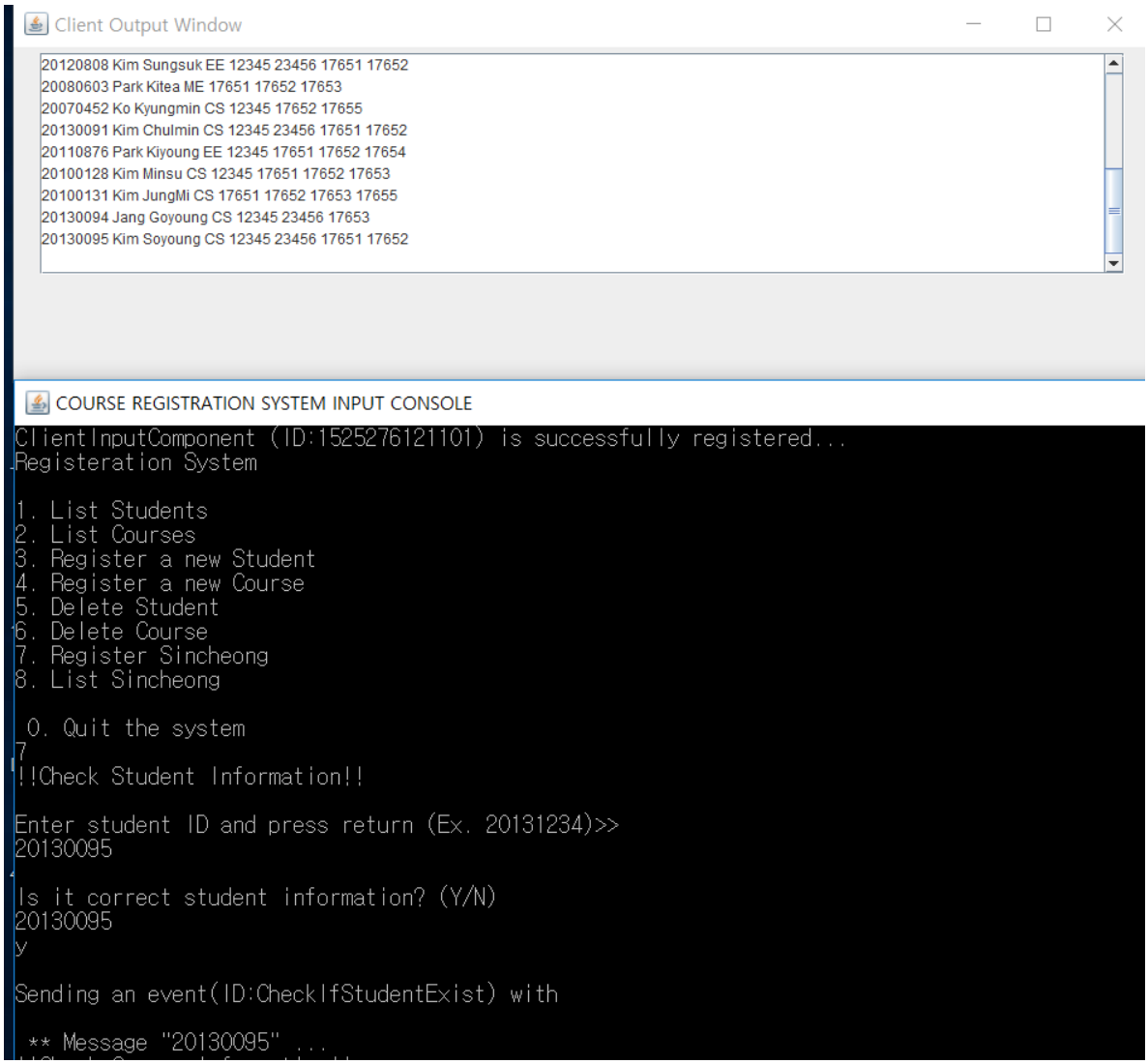


2. 삭제완료 후 과목리스트 출력

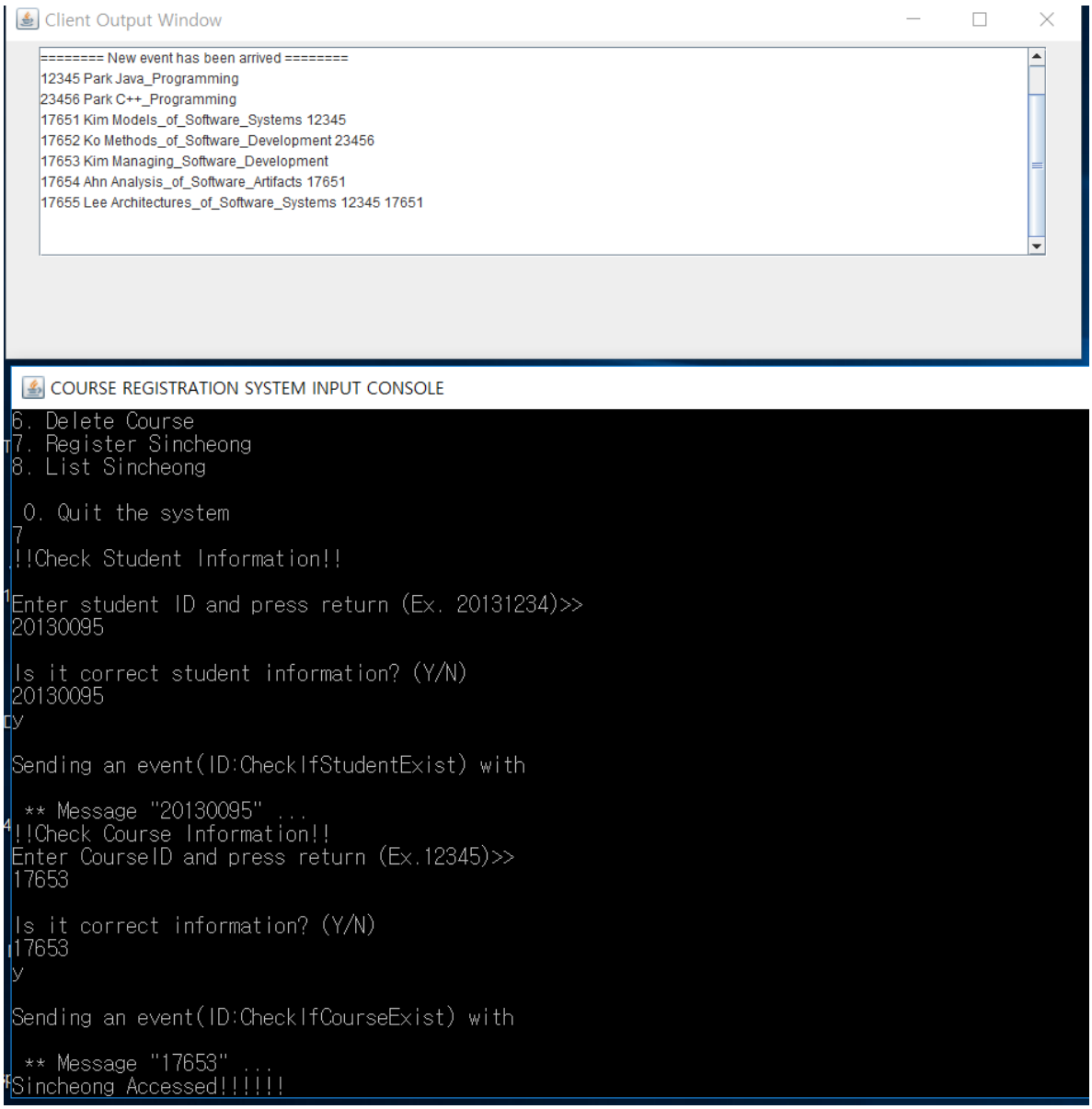


RegisterSincheong

1. 학생ID입력



2. 과목ID입력


Client Output Window
======== New event has been arrived ========
12345 Park Java_Programming
23456 Park C++_Programming
17651 Kim Models_of_Software_Systems 12345
17652 Ko Methods_of_Software_Development 23456
17653 Kim Managing_Software_Development
17654 Ahn Analysis_of_Software_Artifacts 17651
17655 Lee Architectures_of_Software_Systems 12345 17651
COURSE REGISTRATION SYSTEM INPUT CONSOLE
6. Delete Course
7. Register Sincheong
8. List Sincheong
0. Quit the system
7
!!Check Student Information!!
Enter student ID and press return (Ex. 20131234)>>
20130095
Is it correct student information? (Y/N)
20130095
y
Sending an event(ID:CheckIfStudentExist) with
** Message "20130095" ...
!!Check Course Information!!
Enter CourseID and press return (Ex.12345)>>
17653
Is it correct information? (Y/N)
17653
y
Sending an event(ID:CheckIfCourseExist) with
** Message "17653" ...
Sincheong Accessed!!!!!!

3. 신청완료 후 신청 리스트 띄우기