TEAM 7 Final Document

Index

1. Elicitation

2. 클래스 매핑

3. GUI

4. 변경사항

1. Elicitation

건국대 홈페이지 ‘건이네’ 운영진과 협의해봤지만, Requirement 리딩로그에 나온 것 처럼,
일반인(건국대 커뮤니티 게시판)에게 직접 물어봐서는 Requirement를 수렴하기가 어려웠습니다. 그래서 팀원 중 한 명이 평소에 불편했던 점을 몇 가지 꼽아 봤습니다.

**Idea**

1. 과목은 같지만 교수님이 ‘다른 경우’가 있다.
2. 어느 정도 시간표를 완성해 두고, 한 두 과목 추가 시 겹쳤다는 경고가 자주 뜬다.
3. 이런 경우 운이 없으면, 1에서의 ‘다른 경우’ 를 생각해야 한다.
4. ‘다른 경우’로 해결이 되지 않으면 완전히 다른 과목을 골라야 한다.
5. 경고가 뜨고, 시간표를 다시 검색하는 과정이 번거롭다.
6. 하나의 시간표를 완성하기 위해선 1~6 과정을 수 없이 반복 해야 한다.
7. 비우고 싶은 시간대가 있는 경우 더욱 문제는 심각하다.
8. 완성했다고 치더라도, 더 나은(개인 판단) 시간표 구성이 있을 수 도 있다.

이상의 의견을 수렴해 Requirement로 만든 결과

**Requirement num / IDEA num - > 기능**

1/2 -> 시간이 겹쳤을 때의 대안으로 “즐겨찾기 리스트”가 있다.

2/1,3 -> “다른 경우”를 없애기 위해 검색 결과에선 하나로 보여준다.

3/7 -> 비우고 싶은 시간대를 정할 수 있는 기능이 있다.

3/5,6-> Req.1,2,3 를 고려하여 시간표를 “자동생성”하는 기능이 있다.

4/7 -> 개인차가 있으므로, 조건에 맞는 모든 시간표를 출력한다.

위에서 적은 “자동생성”과 “즐겨찾기 리스트”가 TEAM7의 메인 토픽이다. 그 목표는 아래와 같다.

Wish List

Automate 구현을 위해 유저가 원하는 몇 가지 과목을 북마크 하는 기능. Automate를 사용하던

하지 않던, 시간표를 짜기 위해선 선행적으로 채워져야 하는 리스트. 시간표가 겹쳤을 경우, 재 검색하는 불편함을 막기 위한 방편이다. 원하는 과목을 막 채워도 상관없다.

Automated Time Tabling :

사용자가 시간표를 직접 짜지 않고, 원하는 강의와 몇 가지 시간 조건만 지정하면, 조건에 맞는 모든 시간표를 보여준다. 여러가지 조건을 지정하더라도, 개인의 취향에 따라 선호하는 시간표가 다르므로, 단순히 모든 경우를 보여주는 레벨을 Automate로 정의한다.

2. 클래스 매핑

<Use Case>

1. **과목**을 검색한다.
	1. **검색한 과목**을 출력해 준다.
		1. 원하는 과목을 **위시리스트**에 등록
			1. 원한다면 위시리스트에서 뺄 수 있다.
			2. 수동으로 **시간표**를 짜는 경우
				1. 위시리스트에 등록된 과목을 시간표에 추가한다.
				2. 시간표에 등록된 과목을 삭제한다
			3. 자동기능을 이용하는 경우
				1. **비우고 싶은** 요일을 선택한다
				2. 비우고 싶은 시간을 선택한다.
				3. 만들기 버튼을 클릭한다.

새로운 창에 조건에 맞는 시간표 목록을 출력한다.

과목 - > Course

검색한 과목 -> SearchedList

위시리스트 -> WishList

시간표 -> ScheduleSet

비우고 싶은 -> CheckInfo

3. GUI

**3.1 GUI클래스**

GUI란 Graphic User Interface를 말하는 것으로 그림 위주의 사용자 환경을 말한다.

일반적인 콘솔 프로그램은 프로그래머가 정한 순서대로 실행되는데 GUI환경의 프로그램은 순차적으로 실행되는 것이 아니라 메시지를 통해서 실행하게 된다.

이러한 GUI클래스의 특징으로는 많은 API함수와 미리 정의된 버튼, 프레임 등의 클래스들, 그리고 이러한 클래스들의 상호작용을 위한 메시지 핸들러, 액션 리스너 클래스 등이 있다.

**3.2 시맨틱 갭**

최종적인 클래스 다이어그램이 나오더라도 해당 프로그램을 GUI로 구현함에 따라서 클래스 다이어그램에 GUI의 클래스들이 들어가게 되면서 최종적인 클래스 다이어그램보다 클래스 관계도 더 복잡해지고 많은 차이가 발생하게 된다.

실제로 구현 단계에서 MyActionListener, JPanel등의 클래스가 많이 생성되게 되었는데 구현 단계에서 이러한 클래스 다이어그램과 최종클래스 사이의 갭을 극복하기 위하여 GUI부분만을 담당하기 위한 ScheduleSetFrame, MainFrame등의 클래스를 만들게 되었다. 여기저기에서 GUI클래스들을 쓰는게 아니라 사용자에게 보여주는 화면을 일괄적으로 처리하는 부분을 클래스로 해서 해당 클래스에서만 GUI를 쓰는 식으로 해서 차이를 최대한 극복하려고 노력하였다.

4. 변경사항

**Edge, Graph 클래스 추가.**

시간표 생성 기능을 원활하게 하기 위하여 자료구조 중 Graph기능을 구현하기 위해서

Edge클래스, Vertex클래스, Graph클래스를 추가로 구현하였다.



**DataBase클래스 추가**

웹 파싱을 하는 기능을 MainFrame에 넣으려고 했는데 관련 기능의 구현이 원활하지 못하여

JSQL라이브러리를 사용하여 MSAccess와 연동시키기 위해 만들어진 클래스이다.

