Coffee Machine

Software Engineering

T2 Final

201511260 문 성찬

201511284 이 종빈

201211356 송 원종

STR Feedback

Result

Total Case	31
Passed	25
Failed	4
Unknown	2

1. **Failed Tasks (4)**

CMS_STC_001_003	Menu Input = 1 농도 = 진하게 온도 = 냉 물 = 1000ml 원두 잔량 = 0g 가루 커피 유무 = x 찌꺼기 유무 = x	<커피머신의 메시지> 가루나 원두가 부족하여 커피를 추출할 수 없습니다. 소리 없음	Fail
CMS_STC_001_004	Menu input = 1 농도 = 진하게 온도 = 냉 물 = 0ml 원두 잔량 = 100g 가루 커피 유무 = x 찌꺼기 유무 = x	<커피머신의 메시지>물이 부족하여 커피를 추출할 수 없습니다. 소리 없음	Fail
CMS_STC_001_005	Menu input = 1 농도 = 진하게 온도 = 냉 물 = 100ml	<커피머신의 메시지>물이 부족하여 커피를 추출할 수 없습니다.	Fail
	원두 잔량 = 0g 가루 커피 유무 = x 찌꺼기 유무 = o	소리없음	
CMS_STC_002_001	Menu input = 4 물 = 100ml	<커피머신의 메시지> 물이 부족하여 물청소를 시작할 수 없습니다. 소리없음	Fail

1-1 CMS_STC_001_005는 코드 상에서 비교할 때 등호가 포함되어있지 않아서 Fail이 된 것임을 알 수 있었다.

Before:

```
if(rm->water ⊱ ms->concentration){ //물 이  있 을 때
```

After:

```
if(rm->water ⋞ ms->concentration){ //물 이  있 을 때
```

1-2 CMS_STC_001_003과 CMS_STC_001_004의 경우 소리가 나지 않아서 Fail이 났던 것이고, CMS_STC_002_001은 물이 부족한 것은 맞으나 소리가 나지 않아서 Fail이 났던 것이다. 결과 적으로 소리가 안나 서 task fail 이 일어난 것이다. 소리는 option이였음으로 소리를 나타내는 메시지를 출력하는 것으로 바꾸었다.

Before

```
void water_fail(){
    message = "물이 부족하여 커피를 추출할 수 없습니다.";
}
void powder_n_bean_fail(){
    message = "가루나 원두가 부족하여 커피를 추출할 수 없습니다.";
}
void uncleaned_fail(){
    message = "커피가루 찌꺼기가 남아있어 커피를 추출할 수 없습니다.";
}
```

After

```
void water_fail(){
    message = "삐!! 물이 부족하여 커피를 추출할 수 없습니다.";
}
void powder_n_bean_fail(){
    message = "삐-삐-삐-!! 가루나 원두가 부족하여 커피를 추출할 수 없습니다.";
}
void uncleaned_fail(){
    message = "삐 삐!! 커피가루 찌꺼기가 남아있어 커피를 추출할 수 없습니다.";
}
```

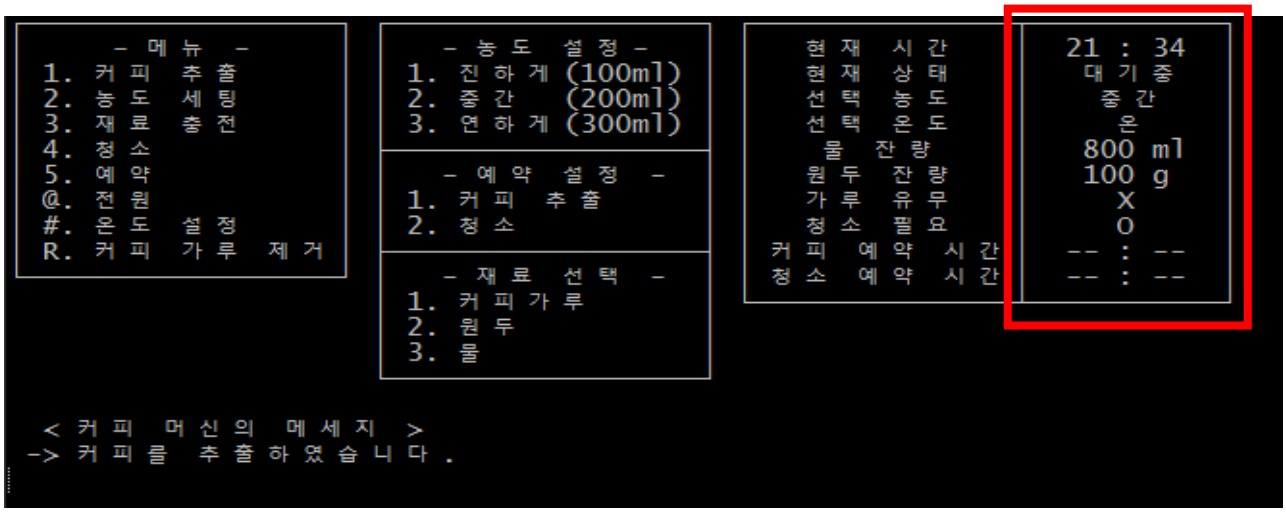
2 Unknown Task

CMS_STC_001_000	Menu Input = 1 농도 = 진하게	<커피머신의 메시지>커피를 추출하였습니다.	Unknown (어떠한 상태의 커피가 추출
	온도 = 냉 물 = 1000ml 원두 잔량 = 100g 가루 커피 유무 = o 찌꺼기 유무 = x		되었는지 확인 불가능)
CMS_STC_001_002	Menu input = 1 농도 = 진하게 온도 = 온 물 = 1000ml 원두 잔량 = 100g 가루 커피 유무 = o 찌꺼기 유무 = x	<커피머신의 메시지>커피를 추출하였습니다.	Unknown (어떠한 상태의 커피가 추출되었는지 확인 불가능)

2-1 CMS_STC_001_000과 CMS_STC_001_002 둘다 unknown에 대한 이유로 커피가 출력 완료되었다면 "커피가 출력되었습니다"가 표시 되지만 어떤 상태의 커피가 나오는지를 알 수가 없다.

이는 실제로 커피가 추출될 때 현재 설정되어있는 커피로 추출되고 나온 커피는 육안 + 맛으로 느끼는 것이다. 또한 우측에 보면 현재 설정되어있는 상태를 확인 할 수 있다.

1) 선택 온도 + 2)선택 농도로 추출되는 커피를 확인할 수 있다.



3 T1의 Feedback & 개선 사항

3-1 Q)예약을 하면 현재 상태가 바뀌지 않음(예약 중), 또한, 다른 행동을 할 수 있었다.

A) 요구 사항의 예약 부분이다.

3.2.6 예약

3.2.6.1 Function

- 사용자는 다음과 같은 동작을 예약 할 수 있다.
  - ■ 커피 추출 예약
  - ■ 청소 예약
- 예약은 버튼 입력과 예약 값 입력을 통해 수행 된다.
- 예약 값은 다음 2 종류가 존재 한다.
  - ■ 예약 선택 (커피 추출, 청소)
  - ■ 시간
- 커피 머신은 예약된 시간이 되면 해당 동작을 수행해야 한다.
  - ■ 커피 추출 예약 → 3.2.1의 커피 추출과 동일하게 동작
  - ■ 청소 예약 → 3.2.5의 청소와 동일하게 동작
- 커피 머신은 다음과 같은 상태일 때 예약을 할 수 없다.
  - ■ 현재 상태가 대기 중이 아닐 경우

3.2.6.2 Input

커피 머신 을 만들면서 가장 고민 되었던 부분 중 하나가 예약 기능이 였다. 실제 커피 기계가 어떻게 작동하는지를 생각 하였고 요구 명시 사항 중 예약 중 "현재 상태가 대기 중이 아닐 경우" 예약을 할 수 없다고 되어있고 예약 도중에 작업을 시행할 수 없다는 구절은 없다. 고로 현재 시각 AM 10:00 일시 AM 11:00 에 커피의 예약을 했을 경우 그 사이 시간 1시간 동안 다른 일을 하지 못하는 것은 말이 안된다고 생각했다. 고로 예약한 시간에 도달 했을 때 작업이 불가능한 것이지 도중에 작업이 불가능하다는 생각은 하지 못하였다.

3-2 Q) 입력하려는 값이 사라져 불편하다. 예를 들어 137을 입력했을 시 1을 누르면 1이 바로 사라짐.

A) 실제로 입력 값이 사라지는 것은 아니고 불편함 외적으로 문제될 것은 아니다. 이는 실시간으로 정보를 받아오는 과정에서 현재 시각을 반영하기 위해 코딩함에 따라 나타난 결과이다.

3-3 Q) 테스트 중, 어느 순간부터 명령을 2번 입력해야 수행하는 상황이 발생함, 커피 추출을 하려고 하는데 대기 상태에서 1을 두 번 입력해야 커피 추출을 수행 하였다.

A) 자체적으로 System Test를 수행 하였을 때 입력을 두 번 해야하는 경우가 없었다. 실행이 느리게 진행되는 경우는 있었지만 실질적으로 입력은 한번으로 충분했다. 위에 질문과 같은 맥락으로 입력 값이 안보여서 두 번 입력을 해야한것으로 느낄 수 있다.

3-4 Q) 추출한 커피가 올바르게 추출 되었는지 확인할 수 없다.

A) 2-1참고

4 Software Engineering을 하며 느낀 점.

-문 성찬

계속되는 SA의 수정과 문서 작업이 매번 압박해왔고, 해당 SA가 제대로 된 것인지 아닌지도 모르는 상황이 너무도 불안했다. 난생 처음 해보는 방식이었다. 프로그래밍을 시작한지도 2년이 되지 않았는데, 설계까지 하면서 하려고 하니 낯설고 어려웠다. 무엇보다도 팀원간의 소통이 제일 중요시되는 작업이었으며, 팀원들을 믿고 의지해야 할 수 밖에 없었다. 설계가 다 되었다고 해서 문제가 해결되는 것도 아니었다. 개발하는 과정에서 쓰이는 data나 process 혹은 controller가 잘못 정의되었거나, 잘못 쓰이고 있다면 그 부분을 다시 처음으로 돌아가서 수정한 후에 개발을 해야 하는 상황도 발생하였다. 때문에 큰 문제를 작은 부분들로 쪼개서 해결하는 SASD는 순차적으로 잘 진행되고, 문제점이 발생하지 않으면서 진행되면 참 좋은 방법이지만 그렇지 않을 경우 계속 앞으로 돌아가서 거기서부터 다시 시작해야 한다는 부분이 많이도 괴롭힌 것 같고, 정확히 알지 못한다면 쓰지 않는 것이 나을 수도 있겠다는 생각이 들게 하는 방법이었다.

-송 원종

SA의 중요성과 코딩과 실제 기계의 작동 간의 괴리감에 고생을 하였다. SA의 경우 처음에 각 프로세스 별, 즉 DFD로 나누는 과정에서 어디까지 포함해야 될지 어디까지 나눠야 할지 그 경계와 input과 output을 컨트롤 하는 것이 어려웠다. 특히 예약 기능을 구현할 때 제한 되는 조건 재료의 유무와 시간에 따라 예약이 가능할지 말지에 대해 생각하는 과정 자체가 힘들었다. 소프트웨어 공학 개론을 처음 접하는 입장에서 SA와 DFD, 유닛 테스트는 매우 생소한 내용으로 데이터의 이동, 컨트롤러의 사용 또한 그 사용법을 정확히 아는데 오래 걸렸다. 소프트웨어 공학 개론의 팀원간의 의사 소통과 역할 분단 또한 매우 중요하게 여겨졌는데 하나에 대해 의논하는 것도 더 좋은 방법이 없나 확인하는 과정, 서로의 의견 교환, 생각을 말로 풀이하는 과정 또한 어려웠다. 무엇보다 모든 작업들이 너무 많은 시간을 차지 하였다. 문서작업을 많이 안해본 입장에서 문서 작업 또한 시간을 많이 차지 하고 계속된 수정 작업은 고되었다. 솔직히 커피 머신 같은 소규모의 작업 시에는 이러한 SA를 만들고 작업을 하는 과정이 필요한가 싶었지만 좀더 커다란 작업을 할 시 효율적인 작업을 위해 꼭 필요한 내용이며 설계 단계에서 확실한 역할 분담과 작업의 세분화는 매우 중대한 역할인 것을 깨달았다. 무엇보다 팀원들간의 소통이 얼마나 중요한지 다시 알게 되었다.

이 종빈

흔히들 개발을 한다고 했을 때 막막했던 부분이, 개발 모델 설정과 팀원들과의 업무 분배였는데 이번 소프트웨어공학개론 강의 중 SASD 과정을 통해 부분적으로나마 체험을 해볼 수 있었고 특히 SASD중 SA부분은 모든 과정 중 가장 뜻 깊은 과정이었다. SA에서는 DFD를 중점으로 작업했는데 DFD의 간단한 모식도, 프로세스간의 데이터 교환, 컨트롤러가 SD작업으로 넘어와 계획적으로 의도한대로 구현되는 모습을 보니 묘한 뿌듯함이 느껴졌다. 물론 원활히 SD작업으로 넘어가는 중 쉬운 일들만 있지는 않았지만 여러 번 다시 처음으로 돌아와 초기 설계의 중요성을 뼈저리게 느낄 수 있었다. 간단한 소프트웨어라도 유지보수, 모듈화 등 공학적으로 효율적인 면들을 만족하려면 어느 과정이 필요한지 어느 도구를 사용해야하는지 알게 되었던 좋은 경험이었다.