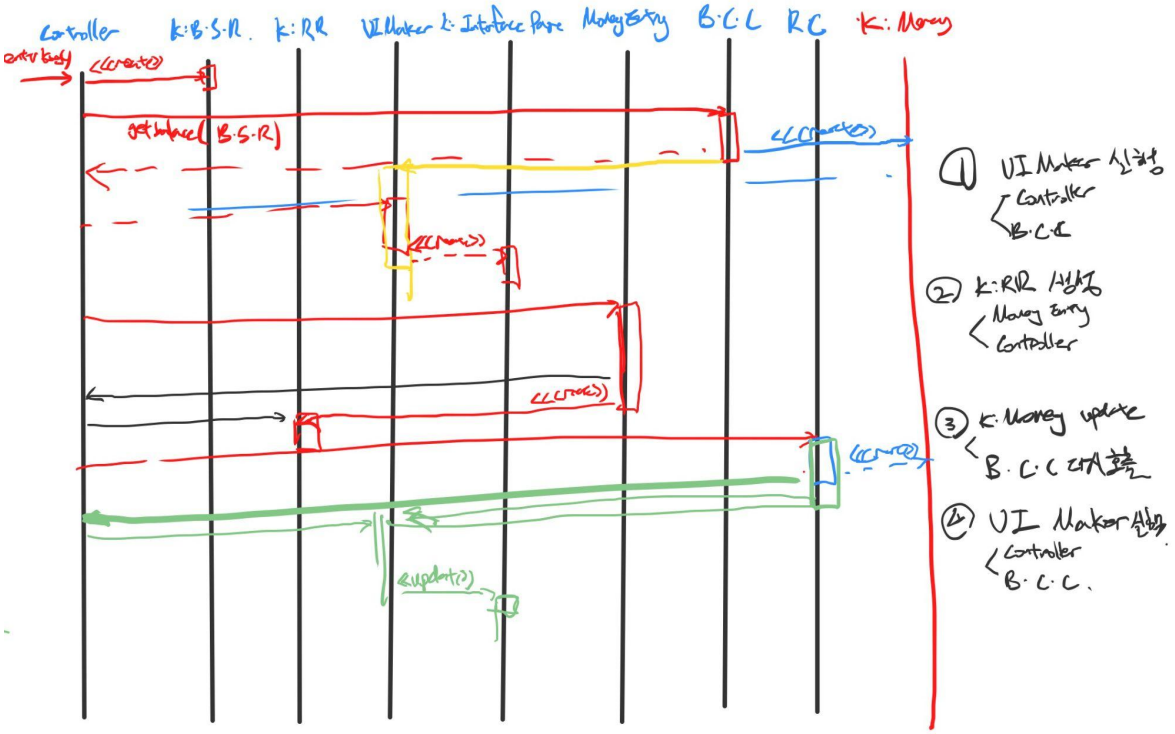
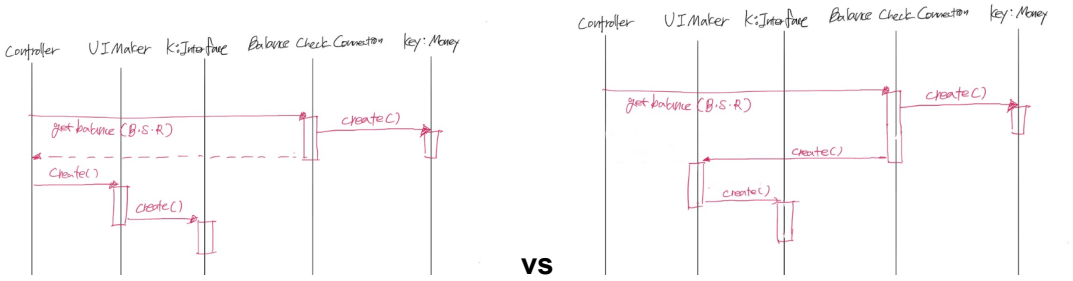
Sequence Diagram

Initial Design



Part1 (case1 & case2)

case1: a) UI maker를 controller가 생성 (b) UI maker을 BCC가 생성



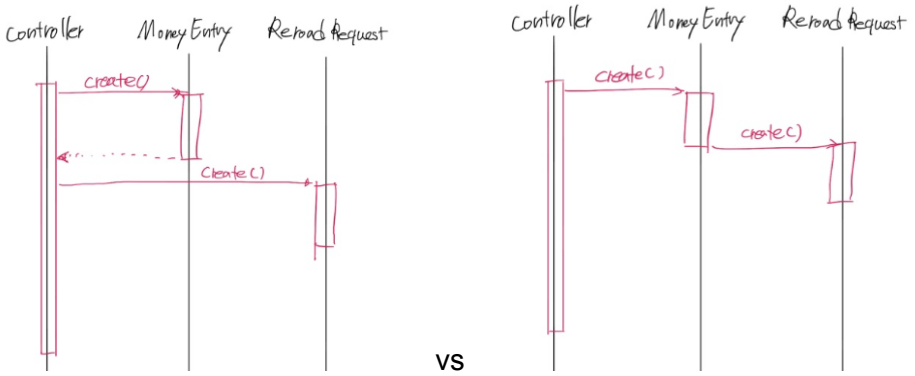
(a) 선택

장점: Balance Check Connection의 strong cohesion

단점: 긴 communication chain

>Balance Check Connection의 기능은 잔액을 조회하는 것이므로 UI maker를 생성하는 기능을 포함시키는 것은 speciality를 해친다고 판단

case2: (a) Controller가 Reload 생성 (b) MoneyEntry가 Reload Request 생성



vs

(b) 선택

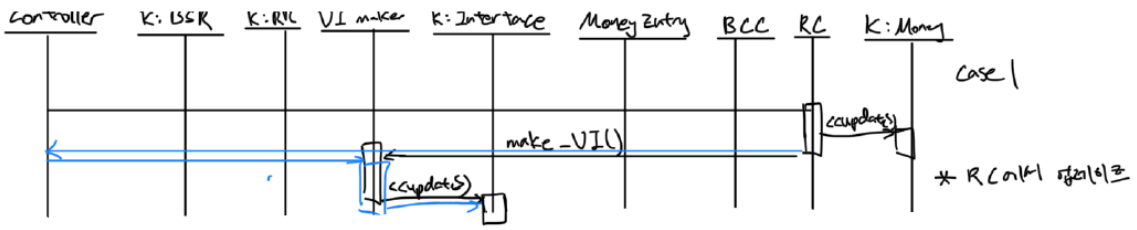
장점: 짧은 communication chain

단점: Money Entry의 strong cohesion

>Money Entry는 충전금액을 입력받는 method로 reload request를 생성해도 문제가 없다고 판단.

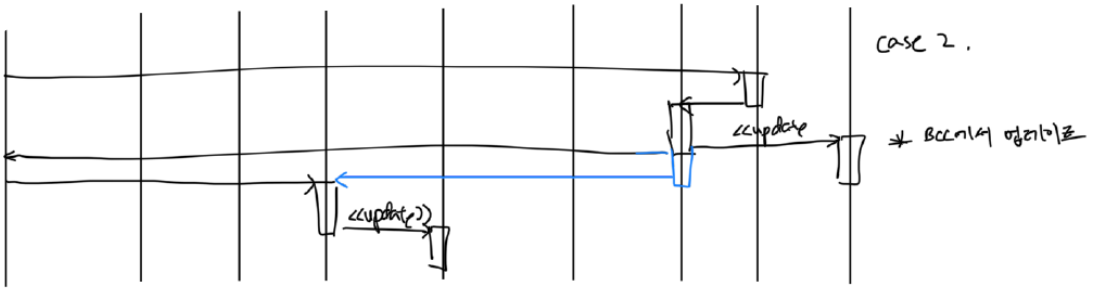
Part2 (case3 & case 4)

case3: Reload Connection이 직접 K:Money를 수정

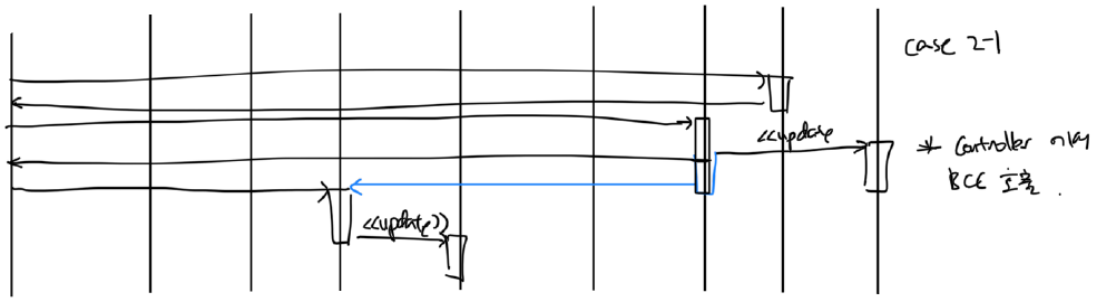


case4: Balance Check Connection을 호출하여 K:Money 를 수정

- case4-0: Reload Connection이 Balance Check Connection을 호출



- case4-1: Controller 가 Balance Check Connection을 호출



case 1 vs case 2

1: Balance check 이 Money 의 변화를 담당한다. (전문성)

2: 레이어의 이동 거리 증가.

case 2 vs case 2-1

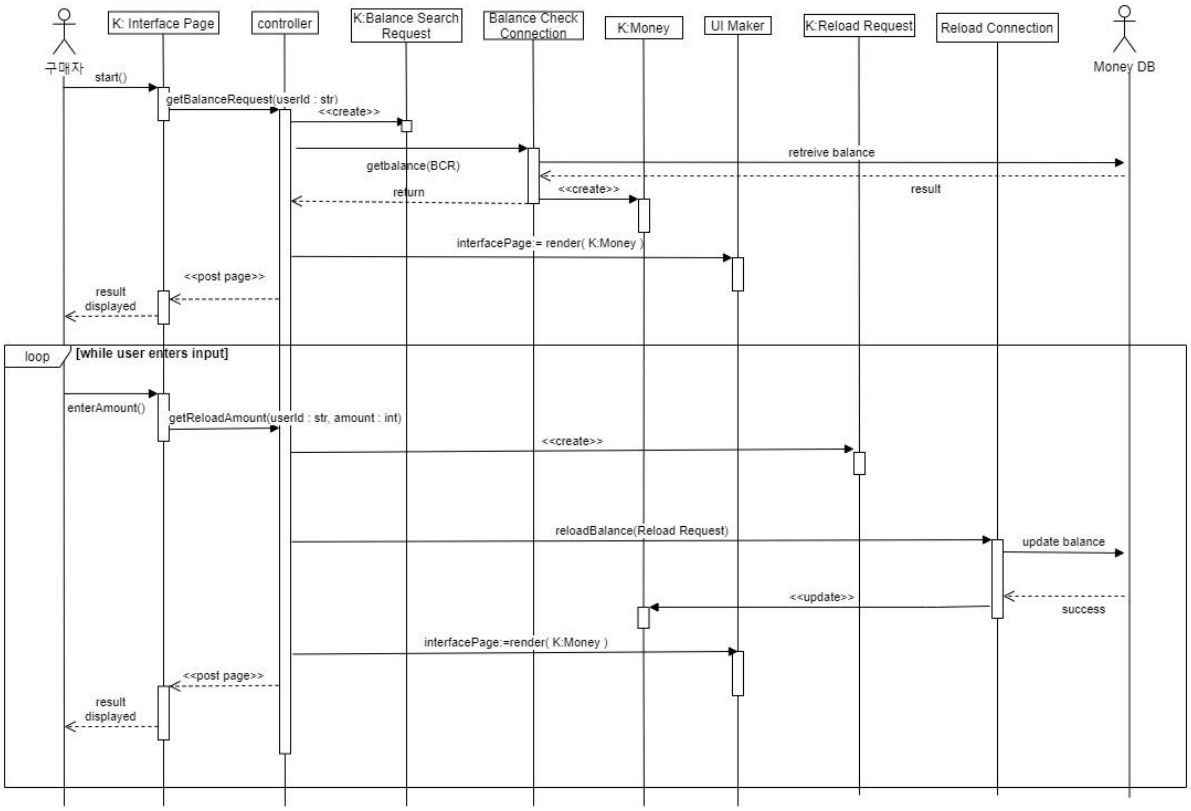
1. BCC 의 호출을 다 같은 level 의 controller 에서 모두 담당한다.
   - BCC 와 RC 모두 같은 level. ∴ Controller가 담당하는 것이 맞지 않을까?
   - High degree of connectivity. cohesion(+) connectivity(-)

case3 vs case4

- case4 사용시
  - Balance Check이 K:Money 의 생성및 수정을 담당한다(전문성)
  - 데이터 체인의 길이가 늘어난다.
- **case3** 사용
  - 데이터 체인의 길이의 증가가 더 큰 손실이라고 판단
  - Balance reload connection이 DB상의 Money를 수정하는 작업을 하므로 K:Money를 수정하는 역할을 하는 것이 맞다고 판단.

Change in final draft

1. interface Page 에서 사용자에게서 입력을 받아 controller을 실행 시키고 사용자에게 결과를 interface page를 통하여 반환 시킨다.
2. 사용자가 금액을 한번 충전하고 반환 하는 것 보다 종료를 원할 때 까지 반복되는 것이 사용자에게 편리할 것이다.
3. interface Page를 update()함수를 통해 UI Maker와 연결시키지 않고 interfacePage:= render()를 통해 나타냈다.



Cohesion Scales

1. Controller
    a. scale-5: controller 에서 마지막 결과를 얻기 위해 순차적으로 functions들을 호출하여야 한다.
    b. controller의 특성상 프로세스를 전체적으로 조절할 수있어야 한다고 판단하였다. 따라서 하나의 function보다 여러개의 functions를 호출하는 scale5가 적절하다고 판단하였다.
2. Interface Page
    a. scale-4: 유저의 입력에 대한 정보를 넘겨준다.
    b. interface Page는 유저와 상호작용하는 부분으로 정보를 시각화하고 입력을 저장하는 부분이라고 판단하였다.
3. UI Maker
    a. scale-6: UI maker은 interface Page를 만드는 하나의 함수만을 수행한다.
    b. UI Maker는 UI를 format에 따라 설계하는 부분으로 단일 작업만을 수행하는 것이 좋다고 판단하였다.

4. Balance Check Connection
    a. scale-6: 단 하나의 특화된 함수만을 실행한다.
    b. Reload Connection과 분할하여 더욱 특화된 기능을 준다. 또한 이렇게 분할할 경우 모듈을 수정할 때에(e.g DB의 이동) 원활할 것이라고 판단하였다.

5. Reload Connection
    a. scale-6: 단 하나의 특화된 함수만을 실행한다.

b. Balance Check Connection과 분할하여 더욱 특화된 기능을 준다. 또한 이렇게 분할할 경우 모듈을 수정할 때에(e.g DB의 이동) 원활할 것이라고 판단하였다.

Solid Principle 적용

1) S - class가 single responsibility를 가지도록 하기 위해서 Reload connection과 Balance Check Connection을 분리하였다.
2) 현재 구상한 모듈을 generic하게 구현하여 이후 수정이 필요할시에 확장을 할 수 있도록 구현할 예정이다.
3) 실제 구현시 Liskov 원칙에 부합하도록 DB connection들의 parent class를 만들 예정이다.
4) 실제 구현시 필요가 없는 interface를 요구하는 상황이 생길 시 interface를 분할하여 필요한 interface만을 요구하도록 한다.
5) high-level modules(Controller, UI Maker)에서 필요로하는 Interface의 자료형과 구조를 미리 추상적으로 지정하여 lower-level modules들의 수정 시 영향을 받지 않게 한다.