

Inf2C Software Engineering 2015-16

Tutorial 2 (Week 6)

Notes on Answers

1 Class diagram for Lift

See Figure 1 on page 2 for an example solution.

Here we imagine there are N floors. For simplicity there is only one `PositionDisplay` class for both the displays above the doors and inside the lift. One could imagine having more specialised classes for each, as with the buttons.

Objects in the `LiftPosition` class record which floor the lift is closest to, along with whether it very near or at a floor. The setup in the diagram imagine the software system is managing the real-time control of the deceleration and stopping of the lift exactly at floor positions. This is perhaps not very plausible. A more realistic setup would be to have a separate `LiftMotionControl` class integrating both the `PositionSensor` and `LiftMotor` classes which accepts messages like `moveToFloor(floor : int)` and has an operation `reachedDestinationFloor()` to model input events signalling move completion.

2 Sequence diagram for Lift use-case

See Figure 2 on page 3 for an example solution.

This has examples of both approaches noted on the question sheet to handling real time: we assume the `open()` and `close()` operations only return on doors fully opening or fully closing respectively. On the other hand, we have the `lift startUp()` operation returning immediately, and on `reportPosition()` operations on the `PositionSensor` to signal how the lift is moving and when it reaches the destination floor.

Exactly as written, the sequence is not realistic: we only exit the loop when the lift reaches the destination floor, and assume then the lift can be stopped immediately.

3 Class diagram for Shopping List App

See Figure 3 on page 5 for an example solution.

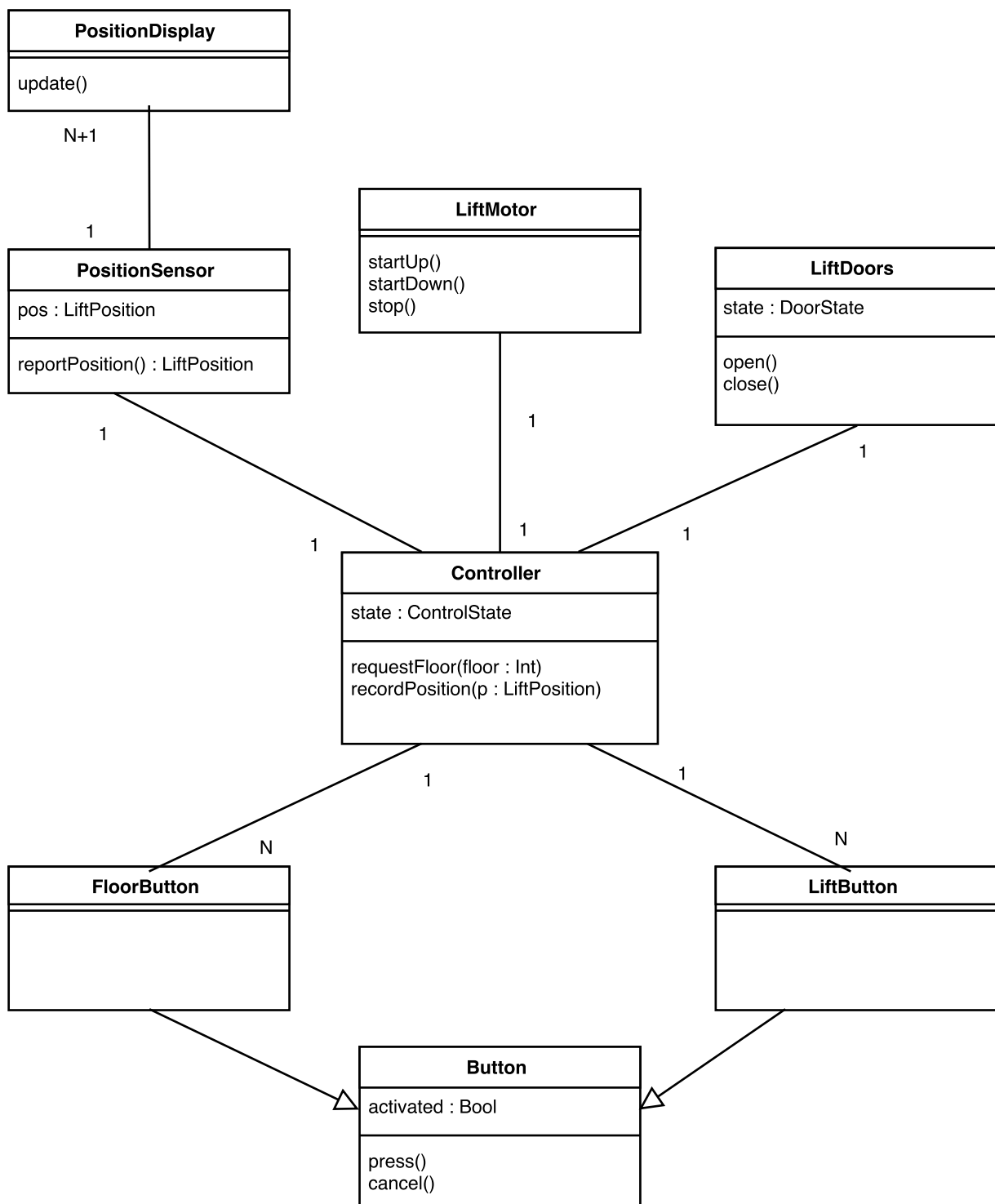


Figure 1: Lift Class Diagram

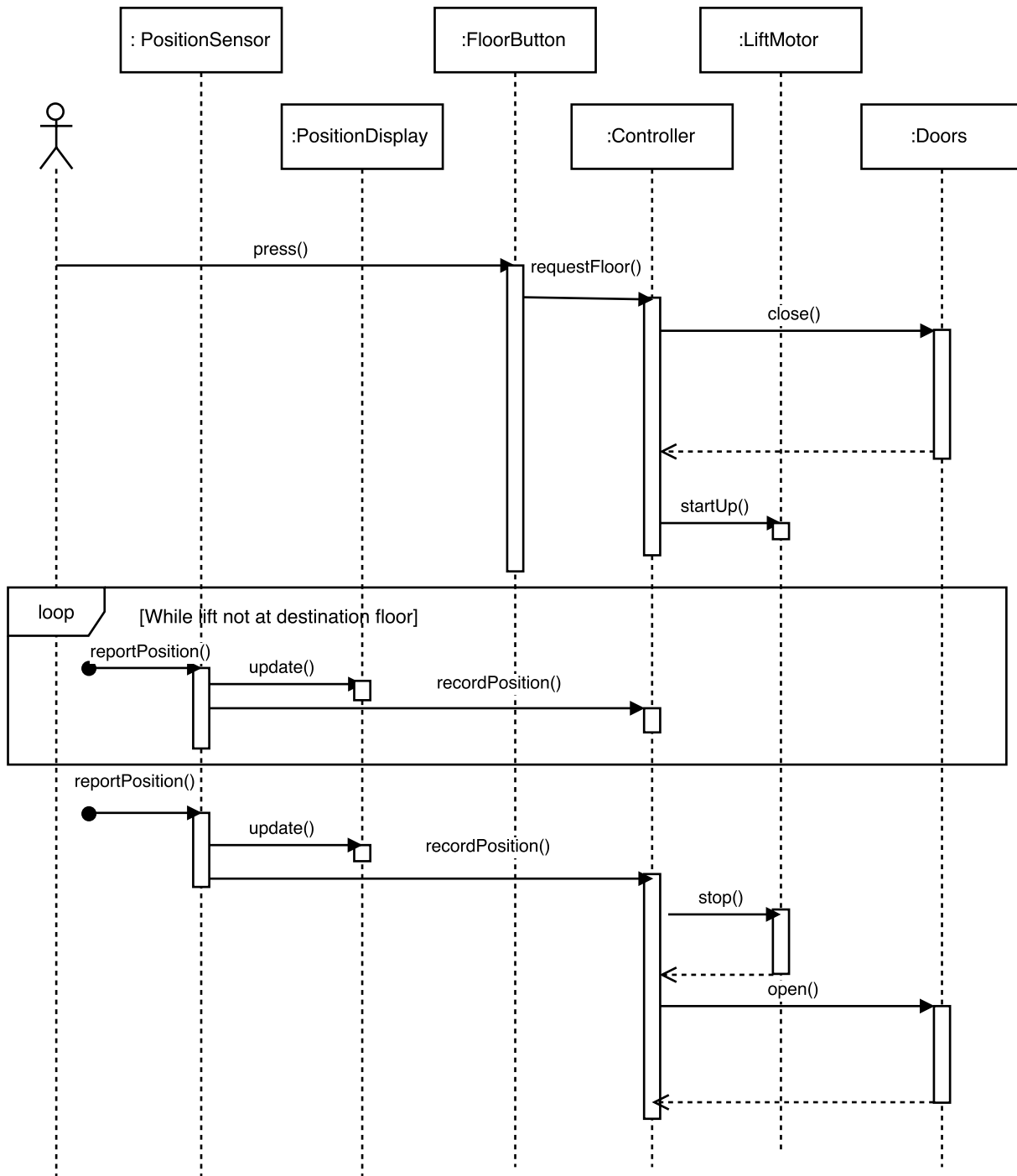


Figure 2: Lift Sequence Diagram

This diagram includes a small amount of plausible inference of some details about how the system must be for it to be sensible. E.g. the identification of the distinct concepts of an Ingredient and an IngredientAmount, and the assumption there are both Ingredient and Recipe libraries.

4 Advanced - Negative Commenting

This depends hugely upon what class diagrams have been produced for the first parts. It might be that your class diagrams are already quite concise, and hence removing each of the classes does indeed have a drastic effect on the system.

In general *negative commenting* on your class diagrams is a useful exercise particularly when first learning to use class diagrams. Hopefully, as you use class diagrams more often it becomes a question you ask of yourself before you *add* a class to the diagram. Hence going over the document at the end is less necessary. But it could rarely hurt.

Paul Jackson. 23rd Oct 2015.

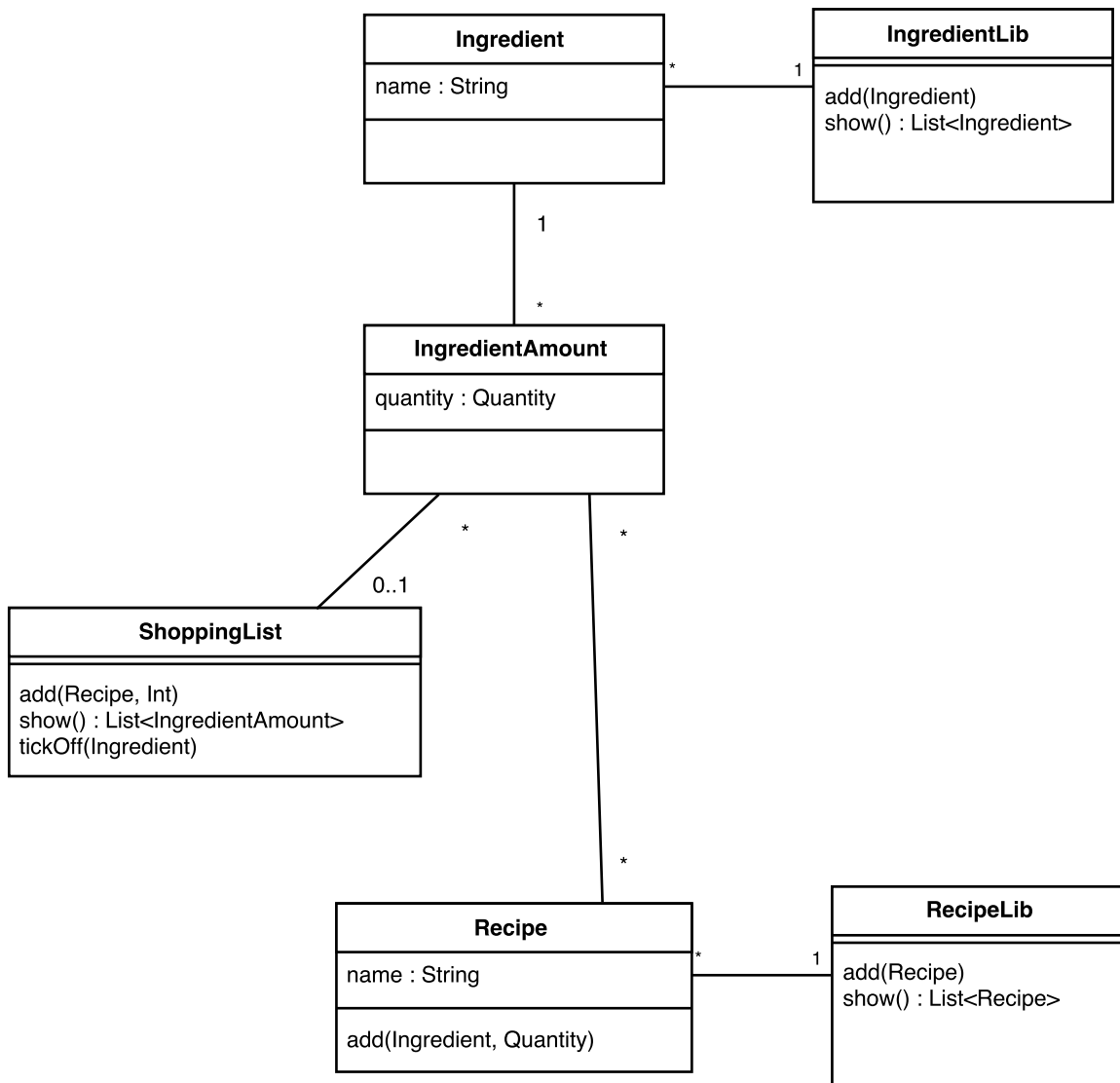


Figure 3: Shopping List App Class Diagram