Exercise: state diagrams

October 9, 2015

Purpose

Let you practise developing state diagrams, first very simple ones and then more complex ones.

Everyone should become able to do these (with some work!). If you have trouble, look at the notes, discuss with one another, ask on Piazza, etc.

1. A party object is created when a venue is booked. An entertainer booking and then a food booking can be added. Before the party takes place a deposit of 20% must be paid. After it takes place, the remaining cost must be paid; once it has been paid, the party object is destroyed.

   Draw a protocol state diagram for the class Party to show the lifecycle of Party objects. Do not include conditions yet (notice whether this causes a problem). Do consider carefully what – in terms of the interface of Party, which you will need to enhance – causes each transition. Remember to include the `getTotalCost()` method that you considered in the Week 3 exercise.

   Draw object diagrams to show how the configuration of objects in the system changes through a typical lifecycle, and list any changes you need to make to the interface of classes in your system, such as new operations for Party.

2. A system developed according to your protocol state diagram is built and deployed, but then the customer reports a catastrophic problem: they are receiving too little money and the system is not complaining! Some investigation reveals that the system is allowing your customer's clients to underpay both the deposit and the balance. Is this permitted by your model? If so, try to fix the problem (i.e., to reduce the likelihood that a system that satisfied your model could have this problem) using conditions on transitions as appropriate.

3. Suppose we wish to show that the party can be cancelled at any time before it takes place. We can use nested states to do this. Show how.

4. In a future release of the software, the processes of booking the entertainer and the food can be done in either order. Use concurrent subregions to show this.

5. What difference would it have made if you had been asked to draw behavioural state diagrams instead of protocol state diagrams? (Hint: not much.)