Protocol Specification Exercises

1. Specify an ebusiness protocol between buyer, seller, and bank in your favorite notation (e.g., UML sequence diagrams, state machines, and so on) that satisfies the following requirements.

   - (In any enactment) *offer* happens first;
   - *accept* and *shipment* happens after *offer*;
   - *instruct* (from buyer to bank) happens after *offer*;
   - *transfer* (of money from bank to seller) happens after *instruct*;
   - An enactment completes whens *transfer* and *shipment* have both occurred.

   (a) Give message schemas for a couple of messages, indicating clearly their senders, receivers, and any information they might contain.

   (b) How flexible is the protocol? Think of all the possible executions supported.

   (c) What communication assumptions does your protocol require to work?

   (d) How might you go about implementing agents (communication endpoints) for the protocol you wrote?

2. Consider interactions between a physician, a pharmacist, and a patient. The patient sends a complaint with the description of their problem to the physician. In response, the physician sends a prescription to the pharmacy and, in turn, the pharmacy lets the patient know that the prescription is fulfilled. Figure 2 depicts the interactions informally.

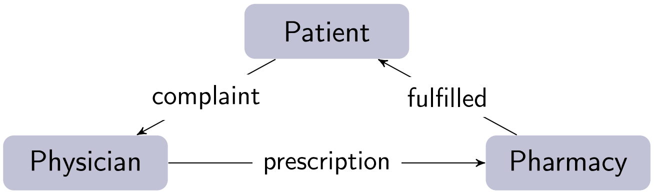

Figure 1: Interactions in a healthcare setting.

   (a) You are given a partially specified information protocol *Healthcare* in `healthcare.txt`. Complete the protocol and verify its safety and liveness.

(b) Now suppose the physician has a choice between sending prescription (to the pharmacy) and advising the patient about how to deal with the problem without the need to take any medicine. Modify the protocol accordingly. Verify safety and liveness.

3. You are given a partially specified information protocol *Ebusiness* in `Ebusiness.txt`. Complete it so that it satisfies the requirements in Question 1.

4. Let's consider test and grading-related interactions between a professor, a student, and a teaching assistant (TA). A professor declares to the student that the test has begun and within the test asks the student a set of questions, one by one. The professor also sends a grading rubric (some specification of how to grade answers) to the TA. The student sends answers to the questions, again, one by one, to the TA, who sends the grade for each answer, one by one, to the professor. Figure 4 shows the interaction schematically and Figure 3 shows an enactment of the protocol.

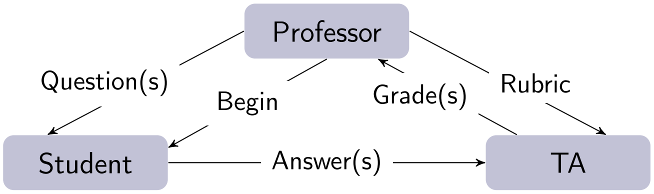

Figure 2: Interactions involved in a test and grade scenario.

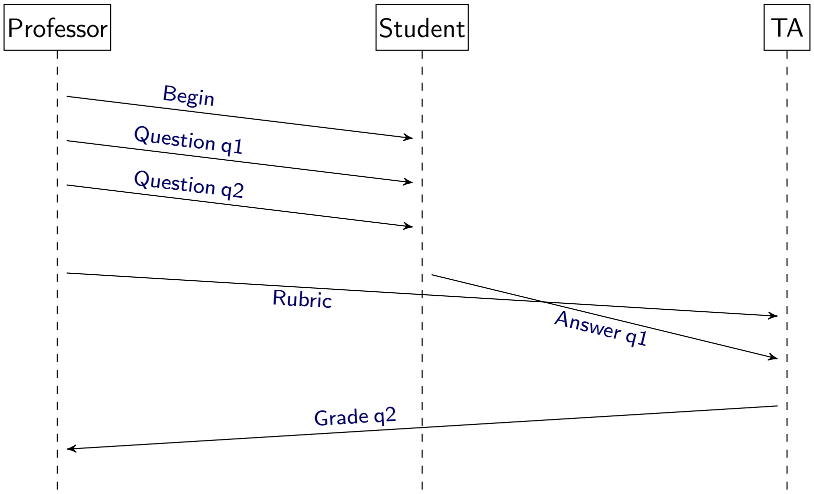

Figure 3: A possible enactment of the desired protocol. The rubric is for the entire test; questions are asked, answered, and graded separately.

(a) You are given the partially specified protocol *Grading* in `Grading.txt`. Fill out the missing pieces in the protocol and verify safety and liveness.

(b) Modify it to take account of the following. The professor can end the test, after which they cannot ask any more questions. The student can resign from the test, after which they cannot send any more solutions. Verify its liveness and safety.

(c) To appreciate BSPL, try to specify the protocol in Part (a) as a state machine. What differences do you observe?

5. Say the content of the local state for a seller agent Alice in the *Ebusiness* protocol were as given in Table 5.

offer(Id:1, Item: wine, Price:4)
offer(Id:5, Item: jamon, Price:6)
transfer(Id:5, Price:6, Details:Urgent, Payment:6)

(a) Give the set of enabled message for Alice.

(b) Write pseudocode for Alice that sends *shipment*s only if *transfer* of the money has occurred. You may assume API that gives you access to the local state and enabled messages.