Exam

Course on UML

2007

Ludovic Apvrille

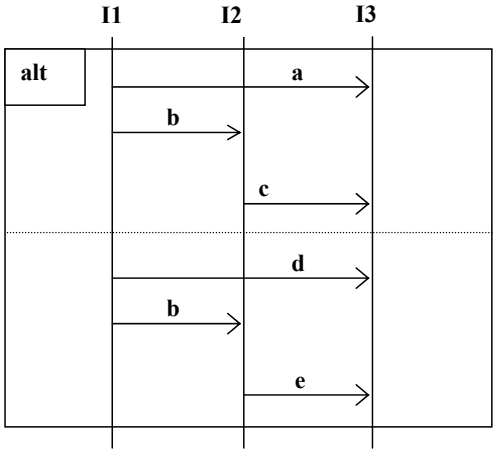
Ludovic.Apvrille@telecom-paris.fr

Authorized documents: courses' slides, notes you've taken during lectures, lab session documents.

Grading is explained for each question. 1 additional point is given for general appreciation (spelling, etc.).
Also, do not spend more than 40 minutes on *exercise I* since the modeling exercise is long.

I. Reasoning Exercise (7 points) ~40mn

We've seen during the last lecture that the language associated to sequence diagrams (or scenarios) has a very large expression power. Thus, an analysis may model a very large set of execution traces, in a very concise way.
This expression power is so large that it is sometimes not possible to make a design that contains exactly the same execution traces. For example, let us analyze the following scenario:

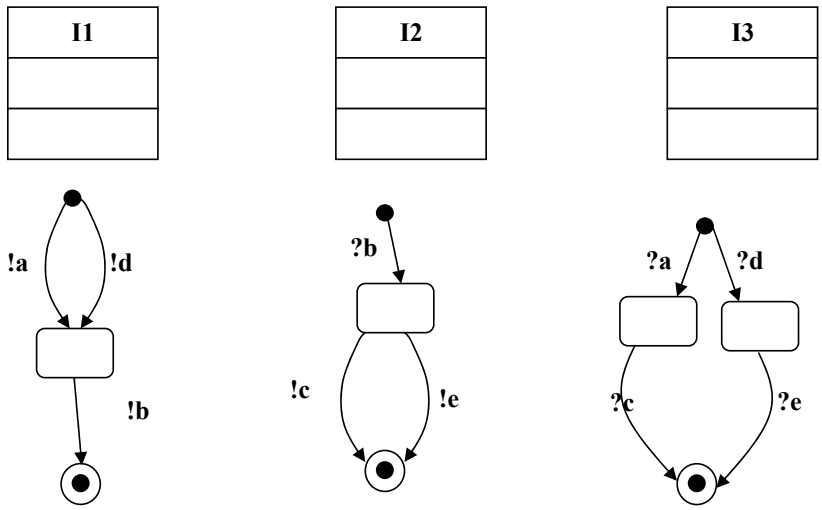


This scenario represents several possible traces, including: (we denote by *!a* the sending of a and by *?a* the receiving of a)
!a !b ?a ?b !c ?c
which means :
I1 sends a, then I1 sends b, I3 gets a, I2 gets b, then I2 sends c and I3 gets c.

1) Modeling an equivalent design

Let's now assume that we're looking for a process P that can generate a design equivalent to an analysis. By "equivalent", we mean that the design has exactly the same traces as the analysis i.e. no more, and no less traces. Process P may be as follows:

- For each instance of the considered scenario, a different class is used. For I1, a class I1 is generated, and so on.
- For each class, a state machine is generated. It represents the execution flow of the related scenario instance. For instance, for I2, the state machine should represent that I2 first gets message b, and then sends either message c or message e.



a) This design has several traces not specified on the scenario. Identify two of them. (1.5 points)

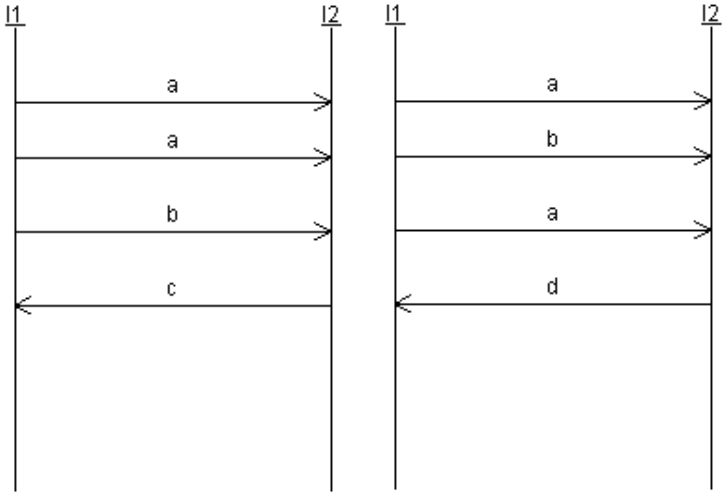
The « equivalent » design having, by process P, more traces than the scenario, and assuming that no other state machines could be found to implement those traces, the scenario is said to be « not implementable ». All design built using process P has at least all traces of the scenario (we won't demonstrate it). In the case of non implementability, it has more traces.

b) Briefly explain why, in your opinion, this scenario is not implementable i.e. the fundamentals behind the fact that process P does not generate an equivalent design. (1.5 points).

2) Implementable or not?

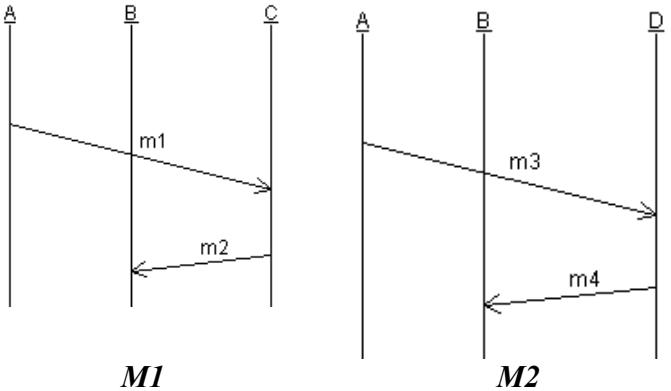
Find out whether each of the following scenario *a)* and *b)* are implementable using process P. It is assumed that there is an alternative between M1 and M2. Grading takes into account the answer only if it is clearly justified, and proved. **No point is given is the answer is not proved,** so, do not try to guess. A good proof should contain the design built using process P, and traces this design has with regards to the ones of the scenario. (2 points for each proof).

a)



M1 ***M2***

b)



M1 ***M2***

III. Modeling Exercise (12 points) ~80mn

The goal is to model the software part of a drone system. The time being short to carry out this modeling, you may omit some modeling details. But if you do so, please, do clearly mention which details you have omitted, and why you have skipped them. At last, do not forget to add comments to your diagrams. Grading takes equally into account both diagrams and comments. If the requirement description of the system is not clear, **do clearly mention your hypothesis**.

A drone is an aircraft with no pilot. It can autonomously take off, fly and land. The drone to model works as follows. It has a remote control system that makes it possible to take pictures. Only the software related to pictures has to be modeled. Pictures can be taken only when the drone is flying. A picture-order is remotely sent to the drone. This order contains the GPS position of the picture to be taken. To know its current position, a drone has an integrated GPS. When a GPS point is reached, with regards to a given margin, the picture is taken, and then stored on a CompactFlash removable storage system. The system needs 2seconds to take a picture, and between 4 and 5 seconds to store it in on the memory card. Pictures may be remotely downloaded, read– taking out the CompactFlash - once the drone has come back from its mission.

1) Analysis ~45mn

a) Make the use case diagram of this machine. (2 points)

b) Make two scenarios, one for the nominal case, and one for a non-regular case. (1 point)

c) Using the technique of "words in the text", propose a collection of classes and objects for this system. (2 points)

d) Refine the two previously performed scenarios. (2 points)

2) Design ~35mn

a) From your analysis diagrams, propose a class diagram containing class relations such as associations, aggregations and so on, and also multiplicity. (1.5 points)

b) Perform the composite structure diagram of this system. Your diagram should model communication channels between system entities. (1.5 points)

c) Make the state diagram of the most important / complex class of your system. (2 points)