ADVANCED PROGRAMMING

Laboratory Practice (CAT3L)

Ordinary Call - May 2021

Published on April 08th, 2021

Simulating the operation of a hospital

Part 1: Concurrent Programming

We want to model the behavior of a hospital where patients will go to for a vaccine. The system must have the following characteristics:

- It has a single entrance with a queue.
- The hospital has a RECEPTION through which patients will enter, a VACCINATION ROOM (with capacity for 10 patients), an OBSERVATION ROOM (with capacity 20) and a REST ROOM for healthcare personnel.
- The number of patients the system generates is 2,000. To avoid agglomerations, patients will arrive in a staggered way at intervals of between 1 and 3 seconds.
- In the hospital there will be healthcare and auxiliary workers (both modeled as threads) to care for patients arriving at the hospital.

The behavior to model for each of the elements should be as follows:

Patients

Patients should be modeled as threads and identified as "PXXXX" where X is a unique number (id), such as P0001, P0023, P0178 etc. Patients will have the following life cycle:

- When they arrive at the hospital, they will keep a queue to register at the front desk.
- During registration, an auxiliary will check your data and verify that it is quoted. In 1% of cases, patients are not listed and should leave the hospital.
- If the registration is correct, patient will wait for the assistant to tell you which vaccination desk to go to.
- Once at the vaccination desk, the healthcare worker will proceed to give them the vaccine.

- After vaccination, patients will enter to the first available observation room post, where they must wait 10 seconds and, if no reaction to the vaccine is observed, after that time they will go home.
- 5% of patients are given some kind of reaction to the vaccine, and they have to be cared for by a healthcare worker at their observation post. Once the healthcare worker gives him the go-ahead, he will go home.

Healthcare worker

There will be a total of 10 healthcare workers, which must be modeled as threads, and that will be identified as "HXX", where X is a unique number (id). Example: H01, H02, H10. Healthcare workers have the following behavior:

- When they arrive at the hospital, they will go to the rest room, for which they will take between 1 and 3 seconds.
- Once they are ready, they will go to the first available vaccination post.
- When a patient goes to the vaccination post, they will wait for a dose prepared by the auxiliary. Once they have the dose available, they will proceed to vaccinate the patient, for which they will need between 3 and 5 seconds.
- Once 15 patients have been vaccinated, they will close their vaccination post and take a random time break of between 5 and 8 seconds in the break room.
- If a problem occurs in the observation room, the first healthcare worker to be available, even if they are on a break, will come to attend him, for which he will need a random time of between 2 and 5 seconds.

Auxiliary worker

There are two auxiliaries, which will be modeled as threads, whose IDs will be A1 and A2:

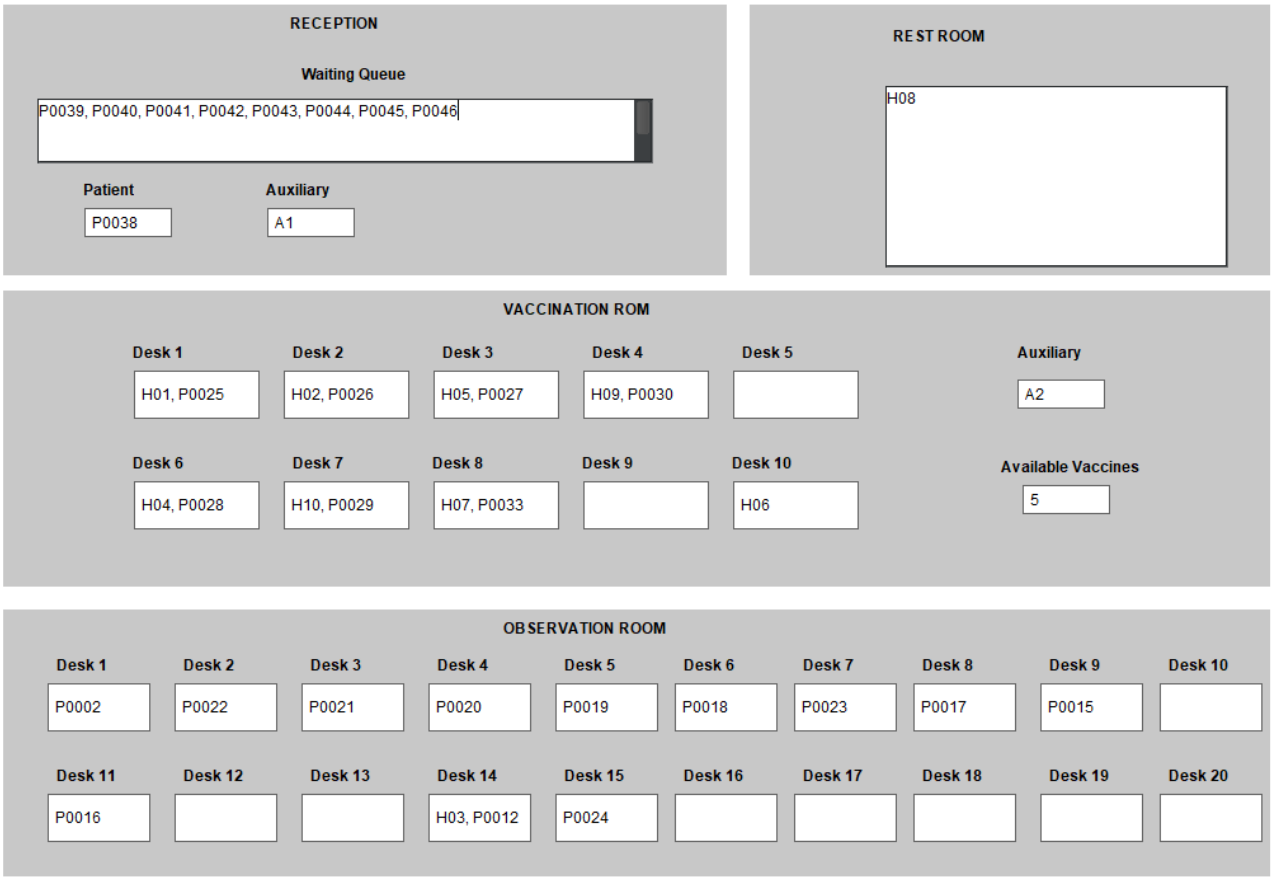
- The first assistant is responsible for checking that the arriving patient is summoned to be vaccinated that day and, if not, will not allow him to pass. In performing this check, the auxiliary takes between 0.5 and 1 seconds. If the registration is correct or, the assistant will check the capacity of the vaccination and observation rooms and if there is a gap then look for an available vaccination desk and inform the patient about the vaccination desk to which to go. In case no position is available the patient should wait. This auxiliary takes a break of between 3 and 5 seconds each time they have successfully registered (successfully or not) 10 patients. The assistant shall keep a record of the patients who have come to the hospital, which shall be shown by at the screen with System.out.println(), indicating the patient ID and vaccination post and the healthcare worker ID who has administered the vaccine. It will also be indicated in the register when a patient has no appointment. An example of the log output is as follows:

```
Patient P0001 vaccinated in desk 5 per H07
Patient P0245 has come without an appointment
Patient P0034 vaccinated in desk 10 per H03
```

Patient P0265 vaccinated in desk 8 per S05

- A second auxiliary is inserting the prepared vaccine doses into a queue, with a rate of 1 every 0.5-1 seconds. This auxiliary takes a break of between 1 and 4 seconds each time you have prepared 20 vaccines.

All system behavior will be saved in a log file (a **text** file called "evolutionHospital.txt"), so that it is easy to analyze what happened. The log will save all the events that are taking place, for example: "Patient P001 vaccinated in desk 5 by H07", "Patient P0245 has come without appointment", "Patient P0012 suffers a reaction and is attended by H03", "H08 begins his rest", "Auxiliary A2 begins his rest", "Auxiliary A1 ends his rest", etc. Each line of that log shall contain the timestamp (date **and time,** including the second particular one in which the event took place) **and the** event itself. One possible example of a system interface would be the one that can be seen in the following image:



It is recommended to vary the different execution times indicated to verify the correct execution of the practice, without problems of race condition, wire leakage, blocking situations, etc.

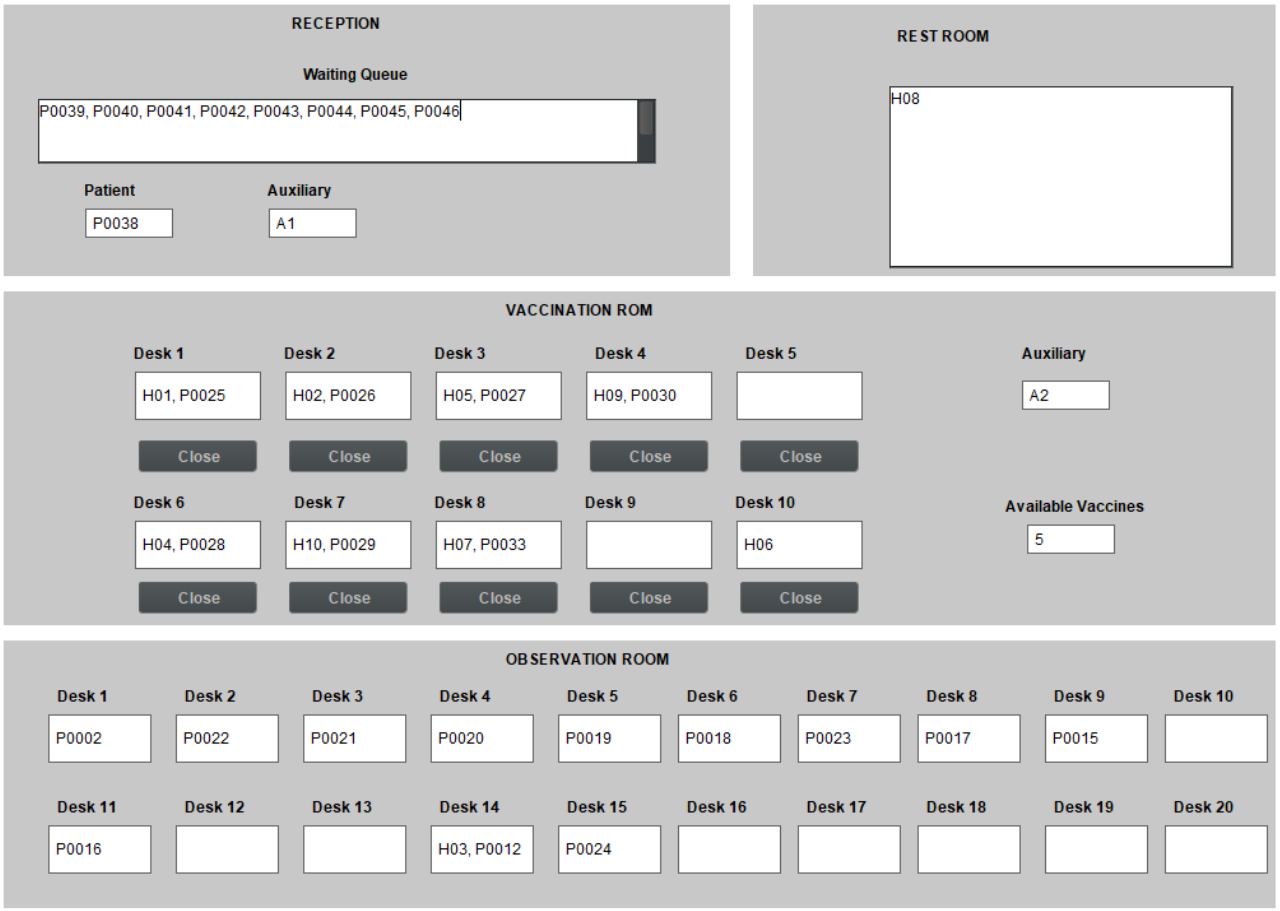
Part 2: Distributed Programming

Based on the previous Part 1, make the necessary modifications to include a new consultation module (using distributed concurrent programming).

This new module will allow to remotely check the status of the hospital. The situation of the different rooms will be updated automatically, that is, without user intervention, updating the interface with a periodicity of 1 second. This interface will also allow to close any vaccination post for cleaning, by means of a button, sending the healthcare worker that occupied it to the rest room. The post will remain closed until it is re-occupied by a healthcare worker who has finished his break.

The solution to this part can be implemented with both RMI and Sockets, at the student's choice.

One possible example of a system interface would be the one that can be seen in the following image:



All mechanisms seen in class may be used to solve all communication and synchronization problems raised in this statement. However, synchronization and communication mechanisms that solve the problem in the most efficient and optimal way possible **should be** used.

Delivery conditions

1. *The practice will be done **(optionally) in pairs** and must be delivered before the date indicated in the Virtual Classroom, through the corresponding task, by uploading two files: the practice memory in PDF or DOC format and the complete Netbeans project, compressed as **ZIP** (don't use the .rar extension) Jobs sent after the deadline for delivery will not be accepted.*
2. *Those practices that does not contain a **NetBeans project** will not be accepted. The use of any other different IDE will imply non-acceptance of the practice.*
3. *If the exercise is done by a couple, **only one of the members must upload it** to the virtual classroom, indicating the name of both students.*
4. ***The report must include, as an annex, the source code** of the program. If this were not the case, the practice cannot be approved.*
5. *The delivery outside the term indicated in the Virtual Classroom will mean a reduction in the final grade, being 25% if it is delivered the next day to the deadline, or 50% if it is delivered within the next two days. Delivery beyond these two days will not be admitted under any circumstances.*
6. ***Both parts** (Part 1 and Part 2) of laboratory practice **shall be delivered together** (i.e. in a single project and a single report), as Part 2 is built on part 1.*
7. *To pass, it is a necessary condition that all programs work correctly and according to the specifications indicated in the statements.*
8. *To approve, the solution must be developed using good programming practices. For example, it is necessary that all the names of the classes begin with a capital letter and all the names of attributes and methods begin with a lowercase letter; the attributes must be private and can only be accessed through getter and setter methods.*
9. *The following data must appear on the cover of the report:*
    a. ***Computer Science Degree/ Computer Engineering Degree***
    b. ***Course 2018/2019 – Ordinary call***
    c. ***DNI – Surname, Name***
10. *The explanatory memory of the practice carried out must include, in the following order: 1) a high-level analysis; 2) general design of the system and the synchronization tools used; 3) the main classes that intervene with its description (attributes and methods); 4) a diagram of classes that show how they are related; and 5) the source code, as an annex.*

11. *Said documentation, except for the code, should not extend more than 20 pages. The quality of the documentation - presentation, structure, content, writing - will be a basic element in the evaluation of the practice.*

12. *For the defense of the practice, if the laboratory teacher considers it necessary, a paper copy of the report, printed on both sides and stapled, must be presented. This document may be used by the student as a basis to answer the questions that arise in the written exercise on the implementation of the application.*

13. *To show the functioning of the programs, it is convenient that each student uses their own laptop, in anticipation of possible problems when installing them in any of the computers of the laboratory.*