Mandatory Assignment 2

INF 2140, Spring 2012

Due: 02.03.2012

Published Date: 17.02.2012

SMALLTOWN'S DENTAL CLINIC

In Smalltown there is a dental clinic where a number of lazy dentists and an efficient secretary offer dental treatments to patients. The clinic is small, so it has a waiting room with a limited number of chairs. The town is well known for the delicious candies it produces, so the clinic has a number of patients who goes to the clinic repeatedly.
During the normal operation of the clinic, one of the following situations can happen:

a) If the waiting room has chairs available, and a new patient enters the clinic, then the new patient sits and waits in one of the available chairs of the waiting room;

b) If there are no chairs available in the waiting room, a new patient cannot enter the clinic;

c) If there are patients in the waiting room and there are dentists available, then one of the waiting patients approaches the dentist and opens his/her mouth, so an available dentist can proceed with his work.

When the dentist has a patient with his mouth opened, he should put the anesthesia before he pulls the sick tooth of the patient (The dentists works with one patient at the time: a dentist can not put anesthesia to one patient and then pull the tooth of another patient). Patients are willing to pay for the service only after the dentist has pulled their sick tooth (so there is not paying in advance). After the current patient has done the payment, the dentist becomes available.

Additionally:

- Since the dentists are very lazy, then the efficient secretary is the one in charge of the correct operation of the clinic. For example, the secretary takes care that the patients do not misbehave in the waiting room; she collects the payment from the treated patient, etc.
- This is a concurrent scenario, so a patient can enter the clinic while the dentists is in the middle of his work.

YOUR TASKS:

FSP modeling using LTSA:

Given the following alphabets:

- `{enter, wait, open_mouth, pay}` for the generic process PATIENT.
- `p[1..P].{put_anesthesia, pull_tooth}` for the generic process DENTIST.

Where `p.1.pull_tooth` means that the dentist pulls the tooth of patient 1.

MODEL WITH ONE DENTIST

1. Model the behavior of the process PATIENT.
2. Using a parameterized composite process, model PATIENTS as an array of processes with

P patients, where P is a constant to fix the number of patients.
HINT: You can use `forall[i:1..P]`
NOTE: The alphabet of PATIENTS is
`p[1..P].{enter, open_mouth, pay, wait}`
3. Model the behavior of the process DENTIST.
HINT: Here you can use indexed processes.
4. Model the process SECRETARY as a monitor with condition synchronization. Which events should happen together?
NOTE: The alphabet of the SECRETARY is the union of the alphabet of PATIENTS and the alphabet of DENTIST.
HINT: You can model the SECRETARY using an auxiliary indexed state process.
```
 SECRETARY = MONITOR[D][CH]
```
Where D records the state of the dentist (value of 1 means the dentist is available, value of 0 means the dentist is busy with a patient) and CH records the number of available chairs in the waiting room.
5. Run the CLINIC system with P=3, D=1 and CH=2.
```
|| CLINIC = ( PATIENTS || DENTIST || SECRETARY ) .
```
6. Check that the CLINIC can generate the following trace:
(Note that the system never terminate, so this is just a part of a possible trace).

p.1.enter
p.1.wait
p.2.enter
p.2.wait
p.1.open_mouth
p.1.put_anesthesia
p.3.enter
p.3.wait
p.1.pull_tooth
p.1.pay
p.2.open_mouth
p.2.put_anesthesia
p.2.pull_tooth
p.2.pay
…

MODEL WITH MANY DENTISTS

7. Model DENTISTS as an array of processes with D dentists, where D is a constant to fix the number of dentists.
NOTE: The alphabet of DENTISTS is
`d[1..D].p[1..P].{pull_tooth, put_anesthesia}`
8. Modify the process SECRETARY to work with many dentists
HINT: Now D records the number of available dentists
9. Run the CLINIC system with P=4, D=2 and CH=3.
10. Check that the CLINIC can generate the following trace:

p.1.enter
p.1.wait
p.2.enter
p.2.wait
p.1.open_mouth
d.2.p.1.put_anesthesia
p.3.enter
p.3.wait
p.2.open_mouth
d.1.p.2.put_anesthesia
p.4.enter
p.4.wait
d.1.p.2.pull_tooth
p.2.pay
d.2.p.1.pull_tooth
p.1.pay
...

Java implementation:

11. Implement a Java Applet project to demonstrate the above described clinic system where:

- Processes PATIENT and DENTIST are implemented as threads PATIENT and DENTIST in Java

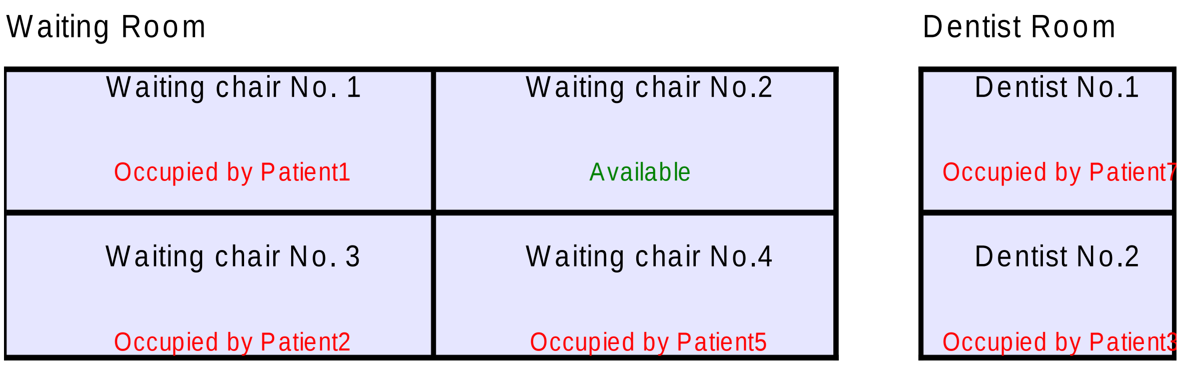
- The process SECRETARY as a monitor SECRETARY in Java.

- The main file (extending Applet) should be named AssignmentTwo.java.

- The graphical interface should display information keeping track of which customer is sitting on which chair and which dentist is doing a treatment on which customer. Hint: For simplicity, students can use StringCanvas class (provided together with the concurrency book) to display that information.

- You may add additional files or Java classes if really necessary.

Below is a proposed graphical representation, but please feel free to make a better one.



12. The implementation should guarantee that there are no race conditions.

13. The implementation should be generic in the following parameters:

- number of chairs in the waiting room;
- number of customers; and
- number of dentists and therefore number of chairs in the dentist room (assuming that number of dentists are equal to number of chairs in the dentist room).

such that they can be changed easily. Note that the graphical interface should automatically adapt when these parameters are changed.

14. The implementation should guarantee a fair protocol which means that a customer who has successfully entered the waiting room, should be served sooner or later.

15. The implementation should guarantee that a patient can leave the clinic only after he has gotten his treatment (i.e., after the assigned dentist has finished his job) and after he has paid.

16. Because we assume that each dentist is dealing with one patient at a time, the implementation should also guarantee that the dentist puts anesthesia and pulls the tooth of the same patient.

17. The implementation should follow Java style conventions for good programming.

DELIVER TO:

The assignment should be carried out individually or in groups of two or three students (Note that groups of more than three students are not allowed) and delivered to the teaching assistant responsible through https://devilry.ifi.uio.no/.
Only one answer is delivered if two or three students work together. The answer must be marked with full name(s) and username(s) of the contributing students.

HOW TO DELIVER:

- You have to implement the FSP using the LTSA Tool (the tool is installed on the lab machines, or you can download it from:
  http://www.doc.ic.ac.uk/~jnm/book/ltsa/download.html).
- You have to make the implementation of the dental clinic in JAVA. We recommend to use the Eclipse tool.
- The delivery must be in one file called “inf2140assignment2.zip” (or “....tgz”). First you copy the contents of your assignment to a folder with the same name as your username (for groups with two or three students, choose one username of the two or three usernames), then delete all compiled files from the java project and pack the folder in a .tgz or .zip archive.
- The project should run with the main file “AssignmentTwo.java”.
  Note: Your final implementation must run when your project is imported into Eclipse, otherwise you must clearly state in the README file the exact commands to run your implementation.
- Your archive should contain
  - The source file of your FSP implementation.
  - The source files of your JAVA implementation (the whole project without the compiled files).
  - A report, named either report.txt or report.pdf, in which you should include the full name(s) and username(s) of the contributing students together with a clear explanation of your FSP and JAVA implementations.
  - A file named README in which you should describe any peculiarities of your solution, for instance, things that may be missing, or assumptions made, or questions for the group teacher. If everything runs fine, state it in the README file.

Evaluation: This assignment is graded pass or fail. You must pass this assignment in order to take the final exam.

Good luck!