Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano

Prof. Elisabetta Di Nitto, Matteo Rossi and Damian Tamburri

20133 Milano (Italia)
Piazza Leonardo da Vinci, 32
Tel. (39) 02-2399.3400
Fax (39) 02-2399.3411

Software Engineering 2 – Written Exam 1 (WE1)

July 1st, 2022

Last name, first name and Id number (Matricola)

Number of paper sheets you are submitting as part of the exam

Notes

1. Remember to write your name and Id number (matricola) on each piece of paper that you hand in.
2. You may use a pencil.
3. Incomprehensible handwriting is equivalent to not providing an answer.
4. The use of any electronic apparatus (computer, cell phone, camera, etc.) is strictly forbidden, with the exception of an ebook reader.
5. The exam is composed of three exercises. Read carefully all points in the text!
6. **Total available time for WE1: 1h and 30 mins**

Question 1 – Alloy (8 points)
Consider a simple elevator, moving people up and down between two floors. In the elevator, there are three buttons: a button 1 to go to the first floor, a button 2 to go to the second floor, and a stop button to stop the elevator. Furthermore, the elevator has three sensors, the first indicating if the elevator is on the first floor or not, the second doing the same for the second floor, and the third indicating if the elevator's door is closed or not. On each floor, there is a button to call the elevator.
The elevator is moved up and down by a motor. The motor is controlled by two signals, one for moving the elevator up and one for moving it down.

Alloy_1 (4 points)
Given the following signatures:

```
abstract sig Bool {}
one sig True extends Bool {}
one sig False extends Bool {}

abstract sig Device {
  on : Bool
}
```

Use them to define a set of signatures and key constraints that capture the elements that constitute the elevator system. Define also a signature that describes the whole system in terms of its components. You are **not** required to define constraints that discipline the behavior of the system.

Alloy_2 (2 point)
Define a fact that specifies that, in an elevator system, it is not possible to give the motor the command to go up if the elevator is on the last floor.

Alloy_3 (2 points)
Define the predicate **safeState** that, given an elevator system, is true if, and only if, the following conditions hold:

- the elevator is not detected by the sensors to be on both floors;
- no command is given to the motor if the doors are open.

Solution

Alloy_1

```
sig Button extends Device{}
sig Sensor extends Device{}

sig Motor {
  goUp : Bool,
  goDown : Bool
}{ not ( goUp = True and goDown = True ) }

sig Elevator {
  b1 : Button,
  b2 : Button,
  stop : Button,
  isf1 : Sensor,
  isf2 : Sensor,
  dooropen : Sensor,
  motor : Motor
}
// properties regarding the behavior of the system, for example the fact that it is
// not possible for both sensors to be "on", are features that one might want to
// verify, so we do not state them as constraints

sig Floor {
  fb : Button
}

sig ElevatorSystem {
  e : Elevator,
  f1 : Floor,
  f2 : Floor
}
```

Alloy_2

```
fact noUpIfLastFloor {
  all s : ElevatorSystem | s.e.isf2.on = True implies s.e.motor.goUp = False
```

```
}
```

Alloy_3

```
pred safeState [ es : ElevatorSystem ] {
  not ( es.e.isf1.on = True and es.e.isf2.on = True )
  es.e.motor.goUp = True or es.e.motor.goDown = True implies es.e.dooropen.on = False
}
```

Question 2 – Function Points (5 points)

DoReMi is a prestigious music school that has several branches in Lombardy, with thousands of students. A software company is implementing DoReMi's enterprise system and is currently focusing on the following functions:

F1. *Candidate student registration*: each candidate student must register to the system, entering email, name, surname, date of birth, his/her preferred instrument, and the corresponding proficiency level.

F2. *Enrolment request*: candidate students can request to enrol in a specific branch of the school. Each candidate can apply for more branches but can send one application at a time.

F3. *Evaluation of enrolment requests*: branch directors can accept or reject the registration requests.

F4. *Visualization of Enrolment Request status*: candidate students can visualize the status of their requests, which can be one of the following: accepted, pending, or rejected.

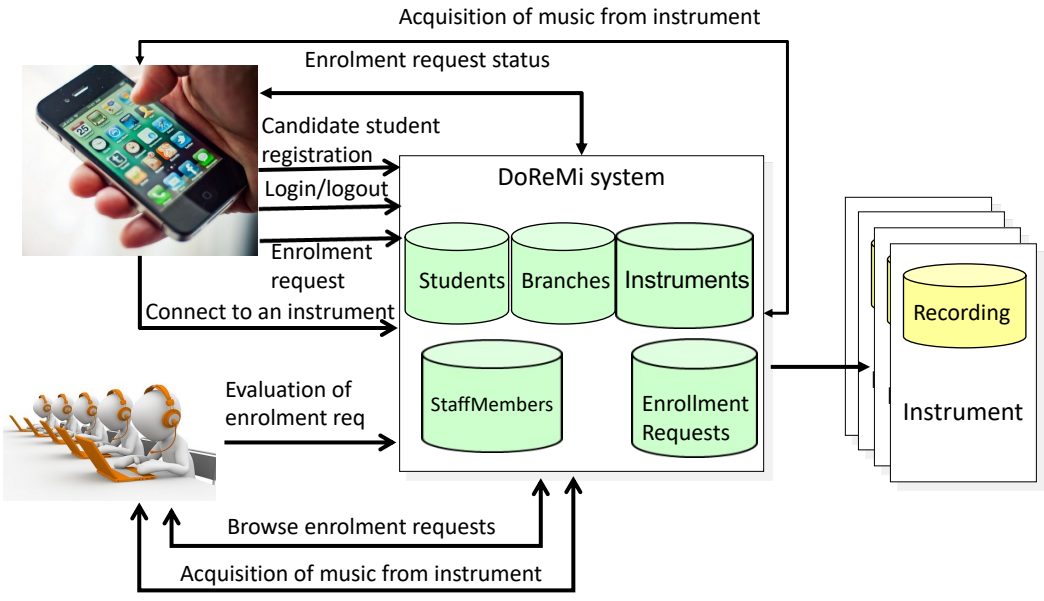
F5. *Connect to an instrument*: enrolled students using electronic instruments can connect their instrument to the system by providing the URL of the instrument's REST interface (assume this is standardized for all instruments),

F6. *Acquisition of music from an electronic instrument*: the owner of an electronic instrument and all teachers in the school can retrieve from the instrument the latest recorded sounds.

Draw a diagram of the DoReMi system in terms of function points and classify them properly as ILF, Ext Inputs, etc. You are not required to analyze the complexity of each function point nor to compute the corresponding count.

Solution

The software to be is going to query the instruments by enrolled students to retrieve the last recordings. For this reason, instruments offer EIFs to our software to be.



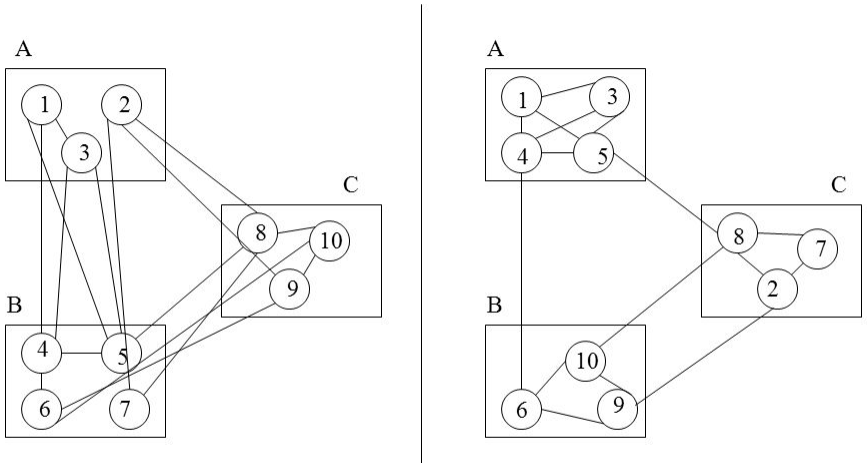
ILF
Student(ID, name, surname, birth date, email, proficiency level, status, preferredInstrumentID, URL): status can be candidate, enrolled; preferredInstrumentID is the foreign key associated with Instrument (see below). We assume that the same instrument, e.g., Fender guitar CD-60SCE Blk WN can be preferred by multiple students and, therefore, represents the complete information about instruments in a different piece of data. The URL is an optional field that is populated by enrolled students with the specific address of a specific instance of the preferred instrument.
Instrument(ID, modelName, characteristics).
StaffMember(ID, name, surname, role): role can be director or teacher.
EnrolmentRequest(ID, studentID, branchID, date, status): status can be pending, accepted, rejected.
Branch(ID, address)

EIF
Recording(date, length, payload)

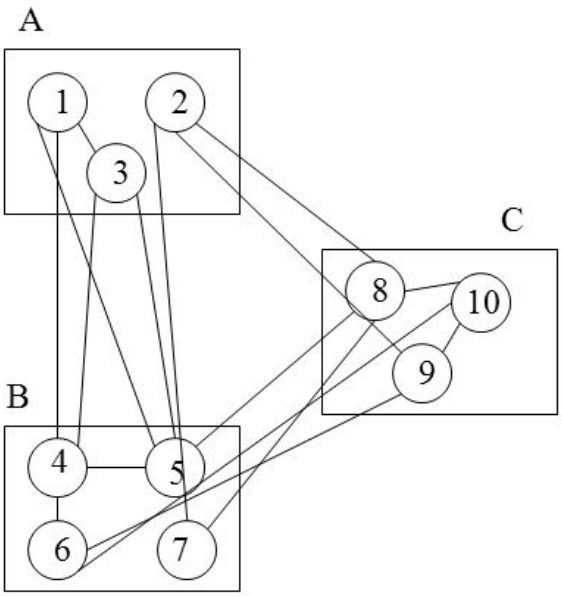
External inputs
Candidate student registration
Login/logout
Enrolment request
Evaluation of enrolment requests
Connect to an instrument

External inquiry
Visualization of Enrolment Request status
Browse Enrolment Requests
Acquisition of music from an electronic instrument

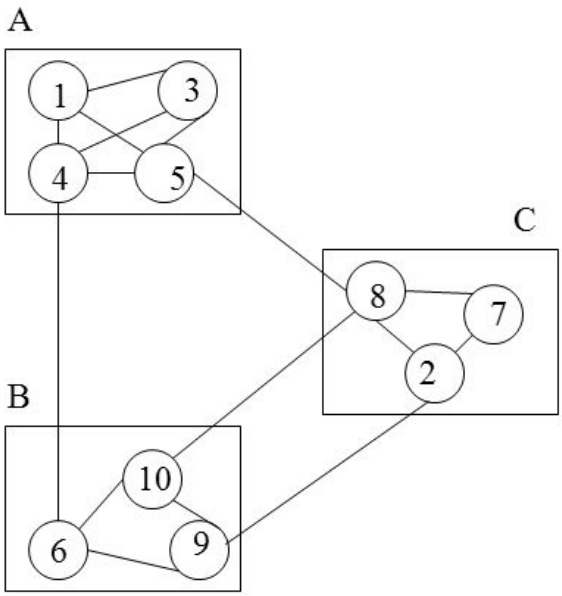
Question 3 – Design Principles in Practice (3 points)
Consider the example architectures below, where boxes represent modules, circles represent functions, and connections between circles represent interactions between functions. The diagram on the left-hand side shows a first version of a system, while the one of the right-hand side shows a second improved version of the same system. Study the figure on the left-hand side, which highlights a problematic condition, and then identify the applied design principles that result in the equivalent redesign depicted in the right-hand side. Illustrate the identified design principles and elaborate on their importance and impact.



Solution



Bad modularization:
low cohesion, high coupling



Good modularization:
high cohesion, low coupling

Low cohesion and high coupling implies that each module is not autonomous but it must rely on the others to perform its tasks. This reduces the reliability of modules and can introduce a significant communication overhead due to the need for frequent inter-module communication.
High cohesion and low coupling, on the contrary, allow modules to be more independent from one another. Within each module functions depend one from the other while communication across modules happens only in specific cases thus limiting its overhead. More independent and cohesive modules increase the possibility of reusing them in other systems and increase their testability.