# Dipartimento di Elettronica e Informazione

Politecnico di Milano

***Prof. Elisabetta Di Nitto, Raffaela Mirandola, Luca Mottola***

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

## Software Engineering II

**February 13th 2014**

| |
|---|
| Last Name |
| First Name |
| Id number (Matricola) |

### Note

1. The exam is not valid if you don't fill in the above data.
2. Write your answers on these pages. Extra sheets will be ignored. You may use a pencil.
3. The use of any electronic apparatus (computer, cell phone, camera, etc.) is strictly forbidden.
4. You cannot keep a copy of the exam when you leave the room.

**Question 1 Alloy (7 points)**

From Wikipedia, an ER model is an abstract way of describing a database. An entity may be defined as a thing which is recognized as being capable of an independent existence and which can be uniquely identified. An entity may represent a physical object such as a house or a car, an event such as a house sale or a car service, or a concept such as a customer transaction or order.
A relationship captures how entities are related to one another. Relationships can be thought of as verbs, linking two or more nouns. Examples: an *"owns"* relationship between a company and a computer, a *"supervises"* relationship between an employee and a department, a *"performs"* relationship between an artist and a song, a *"proved"* relationship between a mathematician and a theorem.
Entities and relationships can both have attributes. Examples: an employee entity might have a Social Security Number (SSN) attribute; the *"proved"* relationship may have a date attribute.
Every entity must have a minimal set of uniquely identifying attributes, which is called the entity primary key. Relations have a cardinality.

For the sake of simplicity, we only consider binary relations, that is, relations between two entities. Moreover, we only consider the following cardinalities:

- *One-to-one*: exactly one entity of the first kind is associated to exactly one entity of the other kind.
- *One-to-many*: exactly one entity of the first kind is associated to more than one entity of the second kind.
- *Many-to-many*: more than one entity of the first kind is associated to more than one entity of the second kind.

**Define in Alloy a specification** that formalizes the structure of ER models according to the above description.

**Define a function that returns** all entities that occur in a relationship with *one-to-many* cardinality.

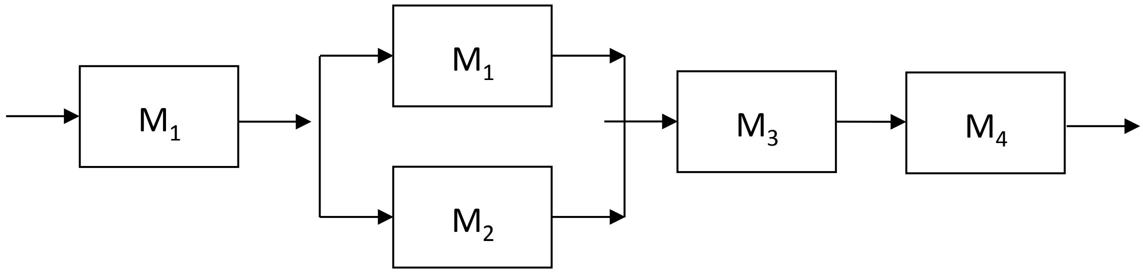
**Solution**

```
sig Attribute {}

sig Entity {
  primaryKey: set Attribute,
  otherAttribute: set Attribute
}{#primaryKey > 0 and primaryKey & otherAttribute = none}

sig Relationship {
  attribute: set Attribute,
  participatingEntitiesSide1: set Entity,
  participatingEntitiesSide2: set Entity
}

sig OneToOneR extends Relationship {
} {#participatingEntitiesSide1 = 1 and #participatingEntitiesSide2 = 1}

sig OneToManyR extends Relationship {
} {#participatingEntitiesSide1 = 1 and #participatingEntitiesSide2 > 1}

sig ManyToManyR extends Relationship {
} {#participatingEntitiesSide1 > 1 and #participatingEntitiesSide2 > 1}
```

```
fun entityInOneToManyR[]: set Entity {
  OneToManyR.participatingEntitiesSide1+OneToManyR.participatingEntitiesSide2
}
pred show[] {#OneToManyR > 1}

run show
```

**Question 2 Architecture (4 points)**

Consider the architecture shown in the figure below.




Knowing that the availabilities of components are the following:

| Component | Availability |
|---|---|
| M1 | 0.99 |
| M2 | 0.98 |
| M3 | 0.98 |
| M4 | 0.97 |

1. Compute the availability of the proposed architecture. Given that you are not allowed to use computers and other devices, we do not expect to see precise calculations, but we would like to see the steps you follow in the calculation.
2. Given the following non-functional requirement: "the system must offer an availability equal or greater to 0.96", explain if the architecture as is allows us to fulfill the requirement and, if not, propose proper modifications. Motivate your answers.

**Solution**
**Point 1**
Let's call $A_5$ the availability of the parallel block composed of $M_1$ and $M_2$.
$A_5=1-(1-A_1)(1-A_2)=0.9998$
$Afin=A_1*A_5*A_3*A_4=0.9409$

**Point 2**
From the calculation above it emerges that the overall availability is lower than 0.96.
In order to fulfill the requirements we can parallelize $M_4$. In this case, the availability of the last block will be $1-(1-A_4)^2=0.9991$, which leads to a total availability of 0.99*0.9998*0.98*0.9991=0.9691.

**Questions 3 Planning (4 points) design (4 point) and testing (3 points)**

We want to develop a system that simplifies the procedures for requiring and obtaining medical services (more specifically, medical examinations) from a hospital.

The system is used by three main actors, the **patient**, the **hospital operator** managing the reservation of the examinations, and the **medical staff**.

When the patient has to require a medical check he/she contacts the reservation service by phone or via web. In the first case the interaction is mediated by the operator. In both cases the goal is to reserve the medical examination for a date that is suitable for the patient and for the hospital.

At the established date the patient goes to the hospital and accesses the system through one of the available self-service points. Here he/she prints the information concerning the medical examination: the name of the doctor, the ambulatory number, the floor where this is located, a number that shows how many persons are already waiting for being examined.

At the same time, the information concerning the patient is transferred to the doctor so that he/she can be ready to receive the patient when his/her turn will occur.

More specifically, the **web system allows each patient to** perform the following operations:

- Require the examination specifying the type and, if possible, a specific doctor.
- Visualize the reply of the system (a reservation).
- Require the modification of a reservation.

Moreover, **it allows the operator to** perform the following operations:

- Visualize the list of doctors offering a certain examination in a certain date.
- Reserve the examination for the patient in a certain date.
- Modify the reservation for the patient.

Also, **it allows each doctor** to:

- Login into the system.
- Visualize the list of patients that have to be checked.

Finally, **the self-service system at the hospital allows the patient to** acquire the information concerning the medical service as explained above.
Given the above description, do the following:

A) **Calculate the function points** for the system. Refer to the following table to associate weights to the function types:

| **Function types** | **Weights** | | |
|---|---|---|---|
| | Simple | Medium | Complex |
| N. Inputs | 3 | 4 | 6 |
| N. Outputs | 4 | 5 | 7 |
| N. Inquiry | 3 | 4 | 6 |
| N. Internal Files | 7 | 10 | 15 |
| N External Files | 5 | 7 | 10 |

B) **Define the system architecture**. Describe the function of each component and the interaction between components. Moreover, list the technologies you are going to use providing a justification for your choice.

C) **Identify two possible acceptance test cases** for the system, one focusing on functional aspects and another on non-functional ones. While defining these cases specify the preconditions (if any) that should be true before executing the tests, the inputs provided during the test, and the expected outputs. Finally, explain why you think the selected tests are important for the specific system being considered.

You can add any hypothesis or functionality you think is important to consider in this case, provided that you do it explicitly.

**Solution**

*Internal Logical Files:*

- Patient
- Hospital operator
- Doctor
- Examination
- Reservation
- Waiting queue

All of them can be considered simple. Thus, 6*7=42 FP

*External Interface Files*: none as there is no evidence that the system is integrated with another one.

*External inputs:*

- Examination request

3 FP

*Internal outputs:*

- Visualize a reservation
- Visualize the list of doctors able to perform an examination
- Visualize list of patients

4*3 = 12 FP

*External inquiries:*

- Modify a reservation
- Reserve an examination

3*2=6 FP