American University of Beirut

Department of Mathematics

Math 282
Final Exam
(Spring 1996-97)

Name : ____________________ id#______________

G. Jalloul time:2 hrs.

I. (40%)

Following is a summary obtained from a study on reuse. The study involved six very large and long-term projects. The research addressed various phases of the software life cycle. In addition it considered administrative aspects such as team structures, measurement models, development environments etc.

P1

The team members interviewed in Japan made impressive claims about the (1) expected productivity they derived from systematic reuse: reduced development time by 10-50 %, improved productivity by 5-10% and improved quality by 20-35 %. These enhancements were due to (2) 40% reuse on the average. Members of the reuse team were praised on their ability to repeatedly specify and maintain newly developed artifacts (3) that will be reusable on future projects. They based their judgment on the (4) internal qualities of the artifact and on the likelihood that (5) these artifacts would be employed in the companies' future applications.

P2

Some of the reuse occurred during the requirements analysis phase. Most of the reused code consisted of (6) small modules that were supported without any adaptations. The rest were (7) adapted to support new functionalities.

Some of the companies involved in the study focused on developing libraries for specific domains. According to the study reuse librarians collected in addition to modules (8) recurring design structures as well as (9) semi complete applications that consisted of sets of interacting objects that provide well defined set of services.

P3

It was claimed that the division of tasks among staff and the structure imposed on teams assisted the reuse process. In addition to the reuse team there were (10) teams whose implementers followed a design based on a requirement specifications. (11) Other teams were created to facilitate sharing results across projects. (12) Employees of a company were in general organized into layers of authority. However (13) Among members of a team goals were reached by consensus and group leadership was a rotating function. (14) A special team was typically set to develop critical applications for the company. In this team, a leader would set the strategy, identifies critical problems and assign experts to different tasks. Noticeably none of the experts would ever be told the full strategy.

P4

To protect the companies' investment, all products were built on top of layer of a software that acted like a virtual machine and sheltered the applications from changes to the host systems. This (15) approach made it possible to move the applications from one machine to another without any modifications. For legacy systems, it was stated that the (16) main approach in using these systems was to encapsulate the system with a a layer f software via which the services of the system could be accessed. The (17) other approach was not even mentioned.

P5

When asked about the cost of training managers all thought more training was needed and worried that the cost could increase by as much as 20 percent to pay for the increased training.

All in all the saving achieved as a result of reuse was assessed in terms of (18) length and functionality (19) coupling, cohesion and structuredness of the developed systems. Conclusively all companies involved praised themselves on quality products that they attributed to the (20) quality assurance measures they applied.

What are the claims of reuse made in P1

1.

2.

3.

4.

5.

Qualify the type of reuse described in P2.

6.

7.

8

9.

What kind of teams were supported by the companies involved as described in P3?

10.

11.

12.

13.

14.

Qualify the approaches used for implementing systems and inparticular legacy systems as described in P4.

15.

16.

17.

What attributes of a software do the qualities described in P5 measure? Except for (19) identify and briefly describe two measurement methods of these qualities. Are these methods well defined? Explain

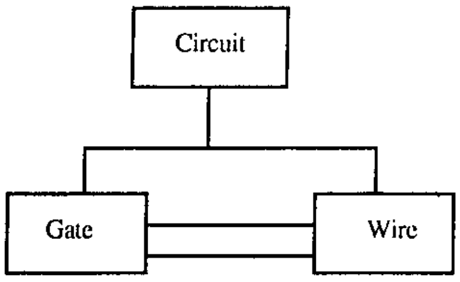
18.

19.

20.

II. (60%)

The object diagram below depicts an object model for digital circuits. A circuit consists of a collection of gates and wires. It inputs one or more signals and outputs one or more signals as well.



1. Discuss two possible implementations of the associations between a wire and a gate. What are the merits of each. In each case modify the object model given above to reflect any additional resulting classes.

2. The interfaces of classes WIRE and GATE are given below. A wire has a signal with (false~0) as an initial value and maintains an array of all gates attached to it. Its behavior is to be supported by feature set_signal as follows: When a wire is informed of anew signal at its input end, it checks to see if the new value is different from the old one. If so, it sets the new value and triggers (call trigger of a gate) all gates connected to it to accomodate the change.

```
Class WIRE
creation
 make
feature
        signal : BOOLEAN;

        make is
              --initializes connections

        set_signal (new_signal : BOOLEAN) is
              -- implements the behavior of a wire described above

        connect_to (g : Gate)
              -- adds gate g to connections

feature {NONE}
        connections: ARRAY[GATE];

end;-- WIRE

class GATE
inherit The_CLock
feature
        trigger is
      -- called by an attached wire when its signal changes

      activate is
           -- called when the delay has elapsed. This procedure defines what the gate
      actually does

end;-- GATE
```

3. Implement feature set_signal of class WIRE

4. The implementation of set_signal as described above includes policy and algorithmic behaviors. Modify the implementation to separate these aspects. Introduce auxiliary features if necessary.

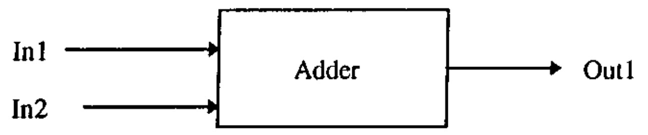
5. All gates in a circuit are driven by a single shared clock. When a gate is informed by a wire that the signal at one of its inputs has changed it sets the clock with its characteristics delay and does nothing further. A clock relies on a scheduler responsible for carrying the computations when the time set for a gate has elapsed. Implement feature clock of class The_CLOCK. Note that class SCHEDULER is not supplied.

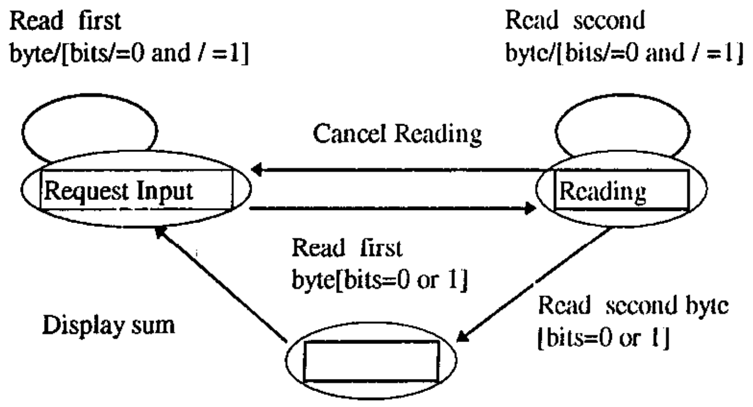
```
Class The_CLOCK
feature
 clock : SCHEDULER is
     -- sharable clock

end;-- THE_CLOCK
```

6. Class GATE is implemented as a descendant of The_CLOCK in order to facilitate usage of feature clock. Modify the implementation of GATE so that it captures the same behavior but using delegation.

7. All Gates require feature activate to implement what a gate actually does. Nevertheless this behavior would differ from one gate to another. Introduce class ABSTRACT_GATE that captures this commonality (feature activate) among gates. Modify interface of class GATE and the given object model to reflect the change.

8. Following is a state transition diagram that depicts the behavior of an 8-bit adder. The adder has 16 input wires that receive the input signals (in1 corresponds to the 8 bits of the first byte and int2 to the bits of the second byte) and eight output wires (out1 corresponds to the 8 output wires) that output the sum of the 2 input bytes (The carry is ignored). The adder can be in state (Request Input) in which case it reads the bytes from standard input consecutively then changes its state to Computing. In this state it computes the sum. Finally it displays the sum. Assuming function do_compute that takes the input wires as parmeters and computes the sum, implement class ADDER.




Read first byte/[bits/=0 and / =1]
Read second byte/[bits/=0 and / =1]
Cancel Reading
Request Input
Reading
Read first byte[bits=0 or 1]
Display sum
Read second byte [bits=0 or 1]

ON YOUR PROJECT

1. List the steps and corresponding commands for building a project in Eiffel from scratch.

2. Draw the object diagram of your project

3. Give the interface of the two most important classes of your project.