Dipartimento di Elettronica e Informazione
Politecnico di Milano
20133 Milano (Italia)
Piazza Leonardo da Vinci, 32
Tel. (39) 02-2399.3400
Fax (39) 02-2399.3411

prof. ***Carlo Ghezzi e Elisabetta Di Nitto***

Software Engineering II
Part I

10 January 2007

Last name
First name
Identification number (Matricola)
Section (specify the professor you are associated with, either Prof. Ghezzi or Prof. Di Nitto)

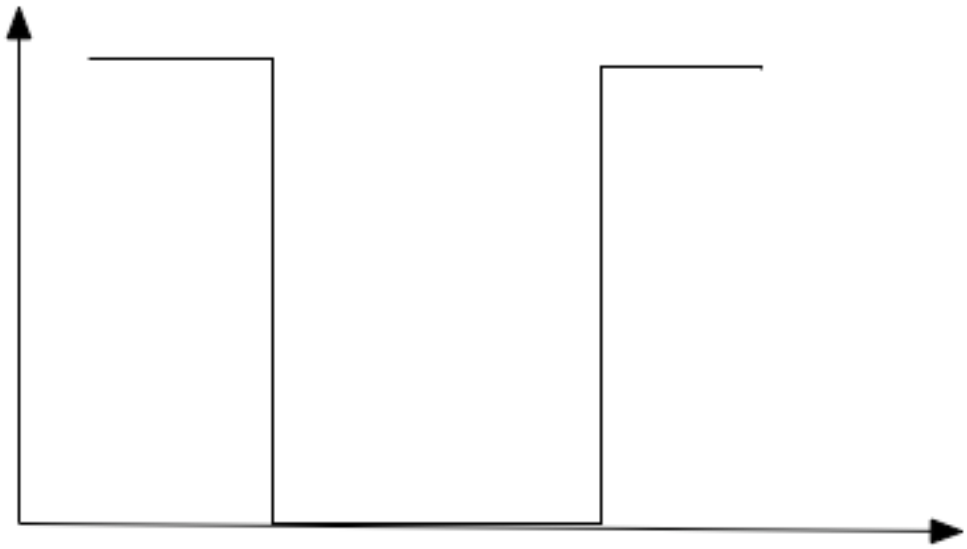
Notes

1. Missing identification data invalidate the exam.
2. Return **only** these pages. Extra sheets will be ignored. You may use a pencil.
3. The exam is in 2 parts. For part 1 you will not be allowed to use books or class notes. For part 2 you will be allowed to use books and class notes. The first part must be in 30 min., the second part in 55 min. The final result is the sum of the scores obtained in both parts.
4. Any use of electronic devices (computers, calculators, cell phones, …) is forbidden.
5. You cannot keep a copy of the exam when you leave the room.

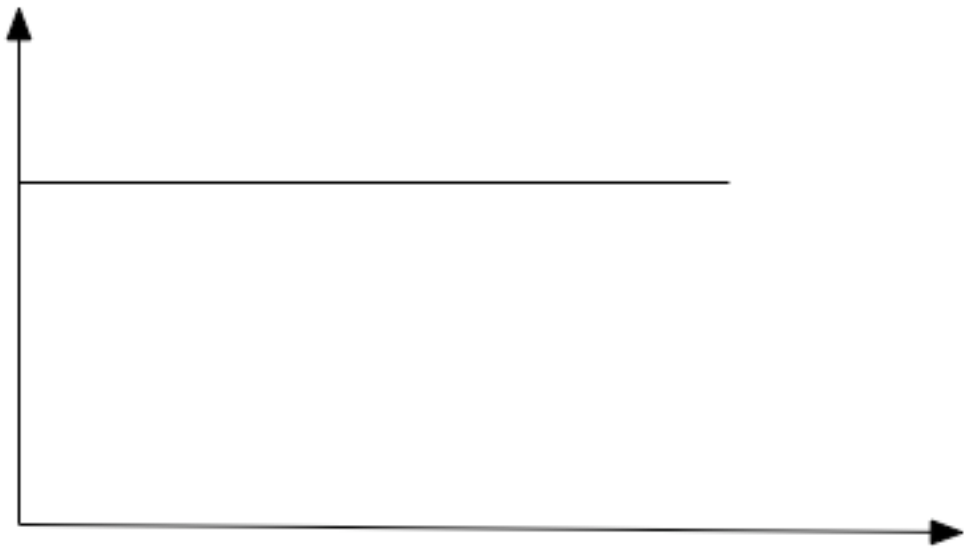
Question 1 – Software development process (2.5 points)

You are asked to draw a diagram that shows the intensity of interaction between the software developer and the customer. Assume that time is on the x-axis and interaction intensity is on the y-axis. Describe how a diagram may look like in (1) the traditional waterfall lifecycle, and (2) a development process adopting extreme programming. Briefly motivate your answer with a few sentences.

The diagram below represents a waterfall lifecycle: Interaction at start (Feasibility/requirements) and at the end (maintenance)



The diagram below represents an XP lifecycle: Interaction uniform along the whole development (customer "on site")



Question 2 – Requirement engineering (2 points)

Explain the meaning of the following concepts in the Jackson-Zave requirements approach:
- requirement
- specification
- domain properties

Also explain which correctness relation must be satisfied among them.

.

Requirements (Req) are what we expect the future system to provide. They are stated in terms of environment phenomena and are prescriptive. Specifications (Spec) are what we expect the interface

between th S2B and the environment to provide. They are expressed in a prescriptive mode in terms of shared phenomena. Domain (Dom) describes the domain knowledge that is relevant for our task. They are in descriptive mode.
Correctess can be stated as Spec and Dom implies Req.

Question 3 – Design (2 points)
What is a publish/subscribe architecture? Briefly describe its main features and its pros and cons.

A P/S architecture expresses coordination among components in terms of message exchange among publishers (senders) and subscribers (receivers). We can distinguish among 3 cases: listener mode (see observer pattern), database trigger mode, and broker mode. They all tend to decouple senders and receivers. In particular, in broker mode, decoupling is managed by the "broker", a middleware layer. Subscribers may subscribe to certain events of interest, publishers may publish them, and the broker does the dispatching.
Pros: high decoupling (no direct bindings), dynamic changes
Cons: quality of service (difficult to enforce delivery policies in terms of message ordering), no persistence.

Question 4 – Testing (2.5 points)
Let T1 and T2 be two test sets that satisfy a criterion C.
Which property do T1 and T2 satisfy if C is consistent?
Which property do T1 and T2 satisfy if C is complete?

If C is consistent either both run without errors or both of them contain one or more tests that reveal an error.
If C is complete this means that if program P has an error, then there is at least a test set that satisfies C that is not successful. Suppose that the program is incorrect. Then, one of the two test sets T1, T2 (or both) may reveal the error, but this does not necessarily happens. We only know that at least one test set exists that reveals an error. So it may happen that both T1 and T2 are successful (which may mean both that the program is correct or incorrect), or either of them or both are unsuccessful.

Software Engineering II

10 January 2007

Last name
First name
Identification number (Matricola)
Section (specify the professor you are associated with, either Prof. Ghezzi or Prof. Di Nitto)

Part II

Question 1 Alloy (6.5 points)

The following Alloy model is a preliminary description of an electronic wallet. It is a card-sized device intended to replace "real" money. Differently from normal credit card, it stores its balance, thus it does not necessarily require a continuous connection to a central server. When a customer buys an item, the amount of payment is transferred from the customer's to the provider's wallet. Of course, the system has to guarantee that this money transfer is executed in the correct way.

```
module ElectronicPayment

// a single coin
sig Coin {}

// a wallet contains a set of coins. Its balance can be calculated by
// summing the amount of each coin
sig Wallet {
 contents: set Coin
}
```

Complete the model above to ensure that both the following properties hold:

1. All coins belong to some wallet.
2. No coin belongs to two wallets at the same time.

Feel free to use all Alloy constructs you need.

Solution

Version 1

```
fact noCoinOutsideWallet {
 all c: Coin {some p: Wallet | c in p.contents}
}

fact noCoinSharing {
 all p1: Wallet {all p2: Wallet {p1!=p2 implies
   (p1.contents & p2.contents)=none} }
}
```

Version 2

```
fact exactlyOneCoinInWallet {
 all c: Coin {one p: Wallet | c in p.contents}
}
```

Question 2 Testing (6.5 points)

Identify two test suites for the function followingDate, the first one respecting the edge coverage criterion and the second one respecting the condition coverage criterion.

```
typedef struct {
      int day;
      int month;
      int year;
} date;
/* definition of boolean data type*/
typedef enum {false, true} boolean;

/* returns true if the parameter is a leap year, false otherwise */
boolean leapYear(int year);

date followingDate(date d)
{
  /* if the month has 31 days */
  if (d.month!=2 && d.month!=4 && d.month!=6 && d.month!=9 && d.month!=11)
     /* if the current day is the 31st, the new day is 1
        (of the following month) */
     d.day = (d.day % 31) + 1;
  /* otherwise if month is not February => month has 30 days */
  else if(d.month !=2)
     /* if the current day is the 30st, the new day is 1
        (of the following month) */
     d.day = (d.day % 30) + 1;
  /* otherwise the month is February. If the year is a leap year */
  else if(bisestile(d.year) == true)
     d.day = (d.day % 29) + 1;
  else d.day = (d.day % 28) + 1;

  /* if the new value of day is 1 we need to update month and
     possibly year */
  if(d.day == 1)
  {
    d.month = (d.month % 12) + 1;
    if(d.month == 1) d.year++;
  }
  return d;
}
```

Solution

Edge coverage criterion
{31/12/1999, 25/04/1999, 02/02/2004, 02/02/1999}
Condition coverage criterion
{31/12/1999, 25/04/1999, 02/02/2004, 02/02/1999, 25/06/1999, 25/09/1999, 25/11/1999}

Note: the instructions containing the module operator hide the existence of a selection statement. For instance:

```
d.month = (d.month % 12) + 1;
```

is actually equivalent to

```
if (d.month == 12) d.month = 1;
else d.month++;
```

The proposed solution does not take this into account and we did not expect from you a solution in this direction.