UNIVERSITÀ DEGLI STUDI DI PISA

Corso di Laurea in Ingegneria Informatica

c/o Dipartimento di Ingegneria dell'Informazione: Elettronica, Informatica, Telecomunicazioni

Esame di Ingegneria dei Sistemi Software

Appello del 29 giugno 2007

Nome e cognome:
Matricola:

Il punteggio in trentesimi associato a ciascuna domanda è indicato fra parentesi.

Scegliere una risposta per ciascuna domanda:

1 In C++ le operazioni polimorfiche sono (1)
- astratte. ☐
- virtuali. ☒
- statiche. ☐

2 Il test funzionale è basato (1)
- sulle specifiche. ☒
- sul documento di progetto. ☐
- sul codice sorgente. ☐

3 Negli Statechart, i segnali sono (1)
- computazioni interrompibili. ☐
- computazioni non interrompibili. ☐
- un tipo di eventi. ☒

4 Un compilatore IDL (1)
- produce codice eseguibile. ☐
- produce codice sorgente. ☒
- produce bytecode interpretabile. ☐

5 Una *formula ben formata* (1)
- è sintatticamente corretta. ☒
- è valida. ☐
- è dimostrabile. ☐

Rispondere alle domande, usando solo lo spazio disponibile:

6 Dimostrare che la formula proposizionale $A \Rightarrow B$ equivale a $\neg A \vee B$. (3)

A	B	$A \Rightarrow B$	$\neg A \vee B$
T	**T**	T	$F \vee T = T$
T	**F**	F	$F \vee F = F$
F	**T**	T	$T \vee T = T$
F	**F**	T	$T \vee F = T$

7 Che cos'è la relazione di flusso nelle reti di Petri? (3)
La relazione che associa posti a transizioni e transizioni a posti:

$$F \subseteq (P \times T) \cup (T \times P)$$

8 Che cosa sono le sequenze (`sequence<>`) nel linguaggio IDL? (3)

Array unidimensionali con un numero variabile di elementi

9 Che cos'è lo studio di fattibilità? (3)
Un'attività volta a stabilire se un prodotto è tecnicamente realizzabile ed economicamente conveniente, a proporre alcune strategie per la sua realizzazione, e a stimarne i costi.

10 Che cos'è un'azione, nel formalismo degli Statechart? (3)
Un comportamento non interrompibile (meno precisamente, istantaneo) associato a una transizione.

11 Con riferimento al codice di Fig. 3, disegnare un diagramma di classi che lo descriva. (5)

12 Rappresentare con uno statechart il comportamento della classe `Prod` **(Fig. 3)**. Considerare `Prod::prod()` e `Prod::dep()` come eventi, `produce()` e `send()` come azioni. (5)

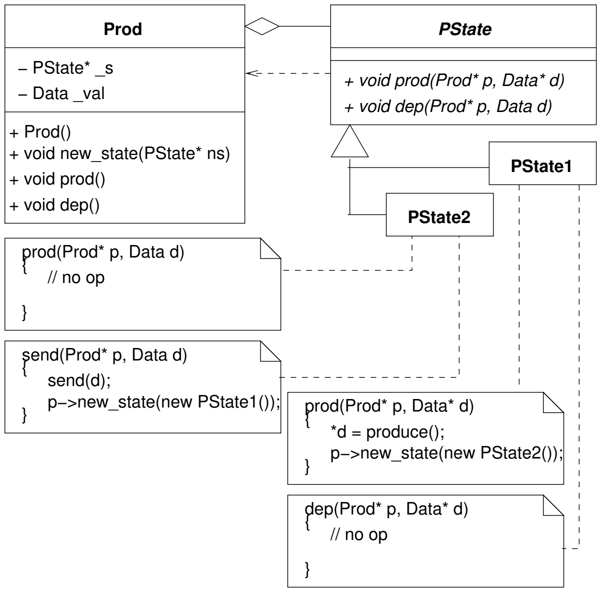

Figura 1: Domanda 11, soluzione.

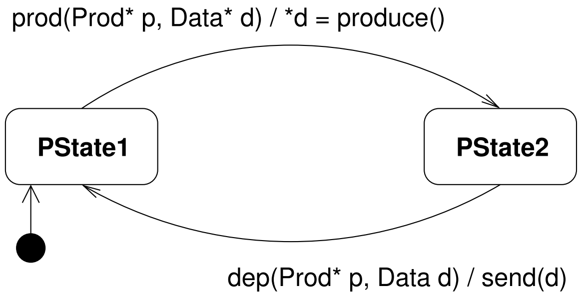

Figura 2: Domanda 12, soluzione.

```
class Prod {
    PState* _s;
    Data _val;
public:
    Prod() { _s = new PState1(); };
    void new_state(PState* ns) { _s = ns; };
    void prod();
    void dep();
};

void Prod::prod()
{
    _s->prod(this, &_val);
}
void Prod::dep()
{
    _s->dep(this, _val);
}

class PState {
public:
    virtual void prod(Prod* p, Data* d) {};
    virtual void dep(Prod* p, Data d) {};
};


class PState1 : public PState {      | class PState2 : public PState {
public:                              | public:
    void prod(Prod* p, Data* d);     |     void prod(Prod* p, Data* d);
    void dep(Prod* p, Data d);       |     void dep(Prod* p, Data d);
};                                   | };
                                     |
void PState1::prod(Prod* p, Data* d) | void PState2::prod(Prod* p, Data* d)
{                                    | {
    *d = produce();                  |     // no op
    p->new_state(new PState2());     | }
}                                    | void PState2::dep(Prod* p, Data d)
void PState1::dep(Prod* p, Data d)   | {
{                                    |     send(d);
    // no op                         |     p->new_state(new PState1());
}                                    | }
```

Figura 3: Domanda 11.