

— Exame —

Desenvolvimento de Sistemas de Informação

LESI/LMCC
2ª Chamada - 2005/06

01/07/2006

Duração máxima: 2h00

Leia o exame com atenção e responda utilizando UML 2.0.

Grupo I

Considere um sistema de inscrições em exames. O sistema deverá ser utilizado por alunos para se inscreverem, pelos SAUM para criarem o calendário de exames, e pelos professores para consultarem os alunos inscritos.

1. Desenhe um diagrama UML em que mostre que:

- tantos os alunos para se inscreverem, como os professores para consultarem os alunos inscritos, executam uma procura de disciplina (para identificarem a disciplina em causa);
- quando o aluno se tenta inscrever, pode acontecer que o sistema negue a inscrição por situação irregular do aluno.

2. Sabendo que na janela da aplicação relativa à consulta de inscritos se pretende que:

- existam duas listas, uma de disciplinas (com todas as disciplinas) e outra de alunos (inicialmente vazia);
- a lista de alunos deverá ser actualizada sempre que o utilizador escolher uma disciplina;
- quando o utilizador selecciona um aluno, é apresentada uma nova janela com a informação desse aluno (e que, para continuar a utilizar a aplicação, o utilizador tem que fechar esta janela).

e sabendo que

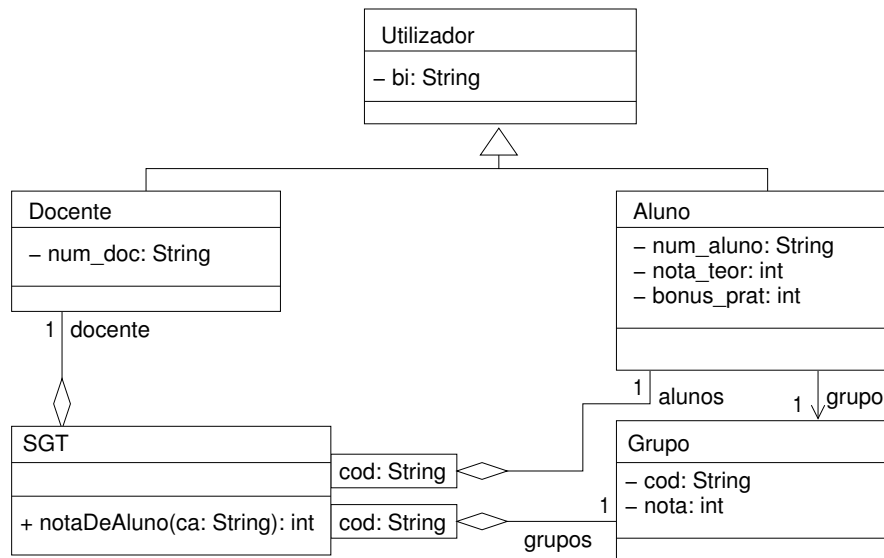
- o método para obter a lista de disciplinas é: `List getDisc()`

- o método para obter os alunos de uma disciplina é: `List getAlunos(cd)`
- o método para obter a informação de um aluno é: `Aluno getAluno(ca)`

Desenhe um **Diagrama de Estados** que modele a sua proposta de interface para este aspecto do sistema. O diagrama deverá identificar os ecrãs da aplicação e seus estados relevantes, as transições entre eles e quando são invocados os métodos da lógica de negócio mencionados.

Grupo II

Considere a proposta de arquitectura apresentada na figura:



1. Sabendo que o método `int notaDeAluno(String ca)` já está implementado, desenhe, com base na arquitectura proposta (e sabendo também que as tabelas são implementadas com Map e as listas com List), um **Diagrama de Sequência** para o método `List<Integer> mediaFilter(int ng)` (da classe SGT) que calcula a lista das médias de todos os alunos com nota superior a `ng` no trabalho (realizado em grupo).
2. Considere que se pretende agora adicionar ao sistema uma tabela de utilizadores. A validação do códigos de acesso deverá ser efectuada pelo método `boolean validaAcesso(String passwd)` de cada utilizador. Os utilizadores do sistema são os docentes e os grupos. Redesenhe o diagrama de modo a incluir esta tabela.

Grupo III

Considere o seguinte excerto de código Java:

```
public class Compras {
    private Map<String,Comprador> baseDados;

    public String eComprador(String bilhete) throws SGCEException {
        boolean encontrado = false;
        Set s = this.baseDados.keySet();
        Iterator i = s.iterator();
        Object chave = null;
        Comprador comprador;
        try {
            while(!encontrado)
                encontrado = testa(i, chave, bilhete);
        }
        catch (NoSuchElementException e) {
            throw new SGCEException("Bilhete nao vendido!");
        }
        return (String)chave;
    }

    public boolean testa(Iterator i, Object chave, String bilhete) {
        Comprador comprador;
        chave = i.next();
        comprador = (Comprador) this.baseDados.get(chave);
        return comprador.comprou(bilhete);
    }
}

public class Comprador extends Entidade implements Registavel {
    private Vector bilhetes;
    private boolean isActive;
    ...
    public boolean comprou(String b) {
        boolean res=false;
        if(this.isActive)
            res=this.bilhetes.contains(b);
        return res;
    }
    ...
}
```

1. Desenhe um **Diagrama de Classes** que represente as classes bem como toda a informação existente sobre elas.
2. Escreva um **Diagrama de Interação** à sua escolha que descreva o comportamento do método `public String eComprador(String bilhete)` considerando apenas a situação em que não ocorrem erros.