deti universidade de aveiro departamento de eletrónica, telecomunicações e informática

Exame de Época de Recurso – 2016-07-12. Duração: 90min.

Questões de escolha múltipla: **responda na grelha**; assinale de forma inequívoca apenas uma opção (se não houver outra indicação, pretende-se a opção verdadeira e, havendo várias verdadeiras, a mais abrangente); as **respostas erradas descontam ½** da cotação; as respostas ambíguas não são corrigidas.

NOME: N.MEC:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A																				
B																				
C																				
D																				
E																				

1.

Considerando o requisito a seguir apresentado:
R_i: O tempo médio entre falhas do componente de autenticação biométrica não pode ser inferior a 120 dias.

a) É um requisito não-funcional, mas inadequado, porque não é verificável.
b) É um requisito funcional, mas inadequado, porque não é parametrizável para assumir outros valores.
c) É um bom exemplo de um atributo de qualidade, relacionado com a facilidade de uso (*usability*) do sistema.
d) É um bom exemplo de um requisito não funcional, relacionado com a fiabilidade (*reliability*) do sistema.
e) Não é um bom exemplo de um requisito, porque não é possível conhecer de antemão o padrão das falhas do componente, antes de ele estar implementado.

2.

A priorização dos requisitos é uma estratégia fundamental em projetos que desenvolvem produtos através de iterações.

a) Os requisitos são priorizados, com a participação do cliente, para ajustar o trabalho a incluir na próxima iteração.
b) A priorização dos requisitos é necessária para escolher a duração que vai ter a próxima iteração.
c) Os requisitos não-funcionais são prioritários e devem ser implementados em primeiro lugar.
d) Os requisitos funcionais são prioritários e devem ser implementados em primeiro lugar.
e) A priorização é feita no início do projeto, identificando requisitos fundamentais e descartáveis.

3.

Na construção de modelos de requisitos, é importante manter a coerência entre três representações funcionais complementares:

a) Diagramas de pacotes, diagramas de casos de utilização e Diagramas de processos.
b) Diagramas de casos de utilização, descrição dos casos de utilização (narrativas), e Diagrama de classes.
c) Diagramas de casos de utilização, Diagrama de atividades e descrição dos casos de utilização (narrativas).
d) Diagrama de casos de utilização, Diagrama de classes e arquitetura candidata.
e) Nenhuma opção anterior está correta.

4.

A análise de requisitos orientada a cenários (de utilização) valoriza a caraterização de funcionalidades que os atores sabem que precisam/querem.

a) Isto é limitativo, porque apenas capta funcionalidades de que os utilizadores se lembram na altura da especificação.
b) Isto é limitativo, porque só utiliza um tipo de diagrama da UML.
c) Isto ajuda a evitar a sobre-especificação de requisitos desnecessários, que podem parecer ser uma boa ideia, mas que ninguém vai usar, porque não estão relacionados com os objetivos dos utilizadores.
d) Isto leva a especificações incompletas, porque não é possível identificar requisitos não funcionais.
e) Nenhuma opção anterior está correta.

5.

A descrição estruturada de um caso de utilização deve incluir:

a) Descrição sumária, decisões e atividades.
b) Estado inicial, transições, estado final.
c) Estado inicial, transições, pós-condições.
d) Pré-condições, fluxo típico, fluxos alternativos.
e) Descrição sumária, decisões, pontos de extensão (*extension points*).

6.

Na Análise, os Diagramas de Classes são adequados para:

a) mostrar as classes de objetos que pertencem ao domínio aplicacional, as suas caraterísticas e relações entre classes.
b) mostrar o modelo de dados para a construção da base de dados;
c) representar as classes que serão necessárias para uma implementação numa linguagem de programação por objetos.
d) ajudar a encontrar a arquitetura candidata da solução de software
e) os diagramas de classes não são adequados para usar na etapa de Análise.

7.

Os diagramas de pacotes podem ser usados para comunicar a arquitetura lógica de um sistema.

a) Mostram como é que vários fluxos num caso de utilização se interlaçam.
b) Mostram quais são os papéis responsáveis por certas tarefas, recorrendo a "empacotamento" dos casos de utilização.
c) Mostram os nós e os componentes envolvidos na instalação da solução.
d) Apresentam módulos e dependências de uso entre eles.
e) Mostram os casos de utilizam como pacotes e os atores como dependências.

8.

Qual o campo lexical mais próprio da fase de Conceção (*Inception*) do Unified Process?

a) Requisitos detalhados, narrativas dos casos de utilização, classes, interfaces.
b) Casos de utilização, implementação exploratória, arquitetura detalhada, componentes.
c) Modelo do domínio, requisitos detalhados, interface do utilizador, componentes.
d) Objetos, classes abstratas, superclasse, subclasse.
e) Modelação do negócio, âmbito, casos de utilização.

9.

Qual das seguintes propriedades **não é** caraterística de um SDLC estruturado e sequencial?

a) Os requisitos do sistema completo estão definidos e documentados à cabeça.
b) Os requisitos são detalhados próximo do momento em que vão ser implementados.
a) O progresso é visto como um fluir direcionado, em que fase anterior desagua na seguinte.
b) É possível voltar atrás e retomar uma fase anterior, apesar de isso obrigar a rever os resultados dessa fase.
c) Os subsistemas desenvolvidos são integrados na parte final da etapa de implementação.

10.

Podemos avaliar a qualidade do desenho de um software por objetos utilizando um conjunto critérios, tais como interdependência (*coupling*) e coesão (*cohesion*).

a) Quanto maior a interdependência, maior é a probabilidade que alterações numa parte do desenho impliquem alterações em outras partes do desenho.
b) Quando um método da classe A refere/utiliza partes internas da classe B há uma situação de interdependência.
c) O princípio da coesão indica que uma classe deve ter um único propósito, e todos os seus métodos estão relacionados com essa finalidade.
d) Todas as opções anteriores estão corretas.
e) As três primeiras opções estão erradas.

11.

Relativamente ao modelo representado no Diagrama 1:

a) Está incompleto: não inclui a fronteira do sistema.
b) Está incompleto: não inclui as linhas de vida dos atores.
c) A "Infraestrutura" não é um ator, porque é um sistema.
d) Deveria incluir associações entre casos de utilização para mostrar precedências relevantes nos processos do negócio.
e) Deve ser suplementado com uma descrição dos cenários subjacentes.

12.

O Diagrama 1 utiliza o conceito de estereótipo da UML (*stereotype*) na relação "extend".

a) É sempre possível substituir o estereótipo "extend" por "include", alterando a ordem na apresentação do diagrama.
b) O estereótipo "extend" serve para especializar o significado da relação geral de dependência.
c) O estereótipo "extend" é acessório, não altera a natureza da relação de dependência.
d) O fluxo de utilização incluído em "Consultar diálogo" inclui sempre a sequência em "Abrir mensagem"
e) O fluxo de utilização em "Consultar diálogo" depende do fluxo incluído em "Receber notificação"

13.

Considerando o Diagrama 2:

a) O diagrama descreve a interação entre objetos de software.
b) O diagrama mostra eventos, fluxo condicional e ações.
c) O diagrama está errado porque não usa partições e as tarefas ficam sem responsáveis.
d) O diagrama tem nós de reunião de fluxos em excesso.
e) O diagrama está errado porque não inclui condições de acesso (*"guard conditions"*) nos nós de reunião.

14.

Considerando o Diagrama 2:

a) É um resultado próprio da etapa de modelação do negócio.
b) É um resultado próprio do desenho do software.
c) Está incompleto, porque não representa os atributos das classes.
d) Permite captar os requisitos não-funcionais para complementar a vista de casos de utilização.
e) Permite representar a arquitetura candidata do sistema.

15.

Considere o processo representado no Diagrama 2:

a) Abrir *ticket*, Atualizar *ticket*, Fechar *ticket* são exemplos de estados (de uma máquina de estados).
b) Quando não é possível Reproduzir o problema, a atividade termina.
c) Depois de Identificar a causa, o caminho a seguir é ambíguo, porque há vários fluxos possíveis.
d) Depois de Verificar a solução, o ticket é fechado.
e) O processo de trabalho é desencadeado por um evento externo.

16.

Considere o Diagrama 3:

a) Um Projeto pode ou não ter um gestor.
b) Um Projeto pode agregar sub-projetos.
c) Uma Tarefa pode agregar sub-tarefas.
d) Uma Tarefa pode ser realizada em diferentes projetos.
e) Cada Colaborador é responsável por uma Tarefa.

17.

O Diagrama 3 utiliza uma classe de associação, para caraterizar a alocação a projetos.

a) A classe de associação é uma classe cujas características descrevem uma associação, e não um objeto "normal".
b) A classe de associação facilita a visualização da interdependência entre duas classes.
c) A classe de associação pode ser suprimida, desde que se mova os respetivos atributos para uma das classes associadas.
d) A classe de associação deve ser usada sempre que há uma multiplicidade de muitos para muitos (entre as classes base associadas).
e) Um bom modelo deve evitar a utilização de classes de associação.

18.

Relativamente ao Diagrama 3, Tarefa e Projeto indicam durações previstas.

a) Há um erro de notação no atributo associado à duração prevista.
b) A duração prevista pode ser determinada à custa de outros atributos, não deve ser representada na classe.
c) A duração prevista pode ser determinada à custa de outros atributos, é um atributo derivado.
d) A duração prevista deveria ser definida apenas na Tarefa.
e) A duração prevista é obtida por um método e é errado apresentar como um atributo.

19.

Relativamente ao Diagrama 3, assinale a afirmação **errada**:

a) Um projeto tem sempre um gestor.
b) Uma Tarefa é um tipo mais especializado de projeto.
c) Uma Tarefa tem sempre um responsável.
d) Um Colaborador por de estar, em simultâneo, alocado a diferentes tarefas.
e) Uma Tarefa pode ser realizada por diferentes Colaboradores.

20.

Considerando a complementaridade entre os tipos de diagramas exemplificados (1, 2, 3);

a) Um Actor pode dar origem a uma Classe, com atributos e responsabilidades.
b) Um método de uma classe pode ser descrito com um diagrama de atividades, para clarificar um algoritmo subjacente.
c) As entidades de dados nos diagramas de atividade podem ser associadas as classes, modeladas num diagrama de classes.
d) Todas as opções anteriores são corretas.
e) Nenhuma das primeiras três opções é correta.

21. [questão de desenvolvimento]

A UML distingue entre modelos estruturais (*structural*) e comportamentais (*behavioral)*.
Qual é o propósito de cada tipo de modelo? Como são utilizados ao longo do SDLC, na Análise e no Desenho?

22. [questão de desenvolvimento]

Considere o trecho de código seguinte, em Java, com omissões.
Apresente um diagrama de classes para representar a informação que se pode inferir do trecho de código.
Procure utilizar toda a informação disponível, revelada pelo código (excluindo as entidades de sistema).

```
public class Draw {
    private ArrayList<GeometricShape> shapesCollection = new ArrayList<>();

    public static void main(String[] args) {
          Draw simpleDraw = Draw.loadFromFile(new File("./mas_draw.dat"));
          GeometricShape shape = new Square(new Point(5, 5), 20.0);
          simpleDraw.add(shape);
          shape = new Circle(new Point(1, 7), 10.0);
          simpleDraw.add(shape);
          System.out.println("Count of shapes in this drawing is "
                      + simpleDraw.size());
          System.out.println("Total drawing area is "
                      + simpleDraw.calculateArea());
    }
[…]
}
```

FOLHA DE DIAGRAMAS

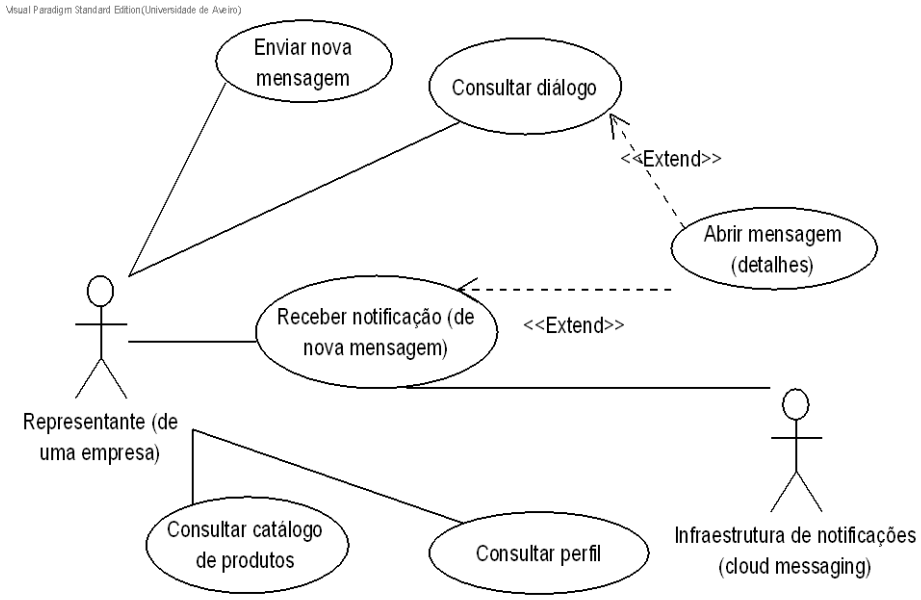

Diagrama 1- Sumário dos requisitos funcionais de uma aplicação móvel para promover contactos comerciais entre empresas.

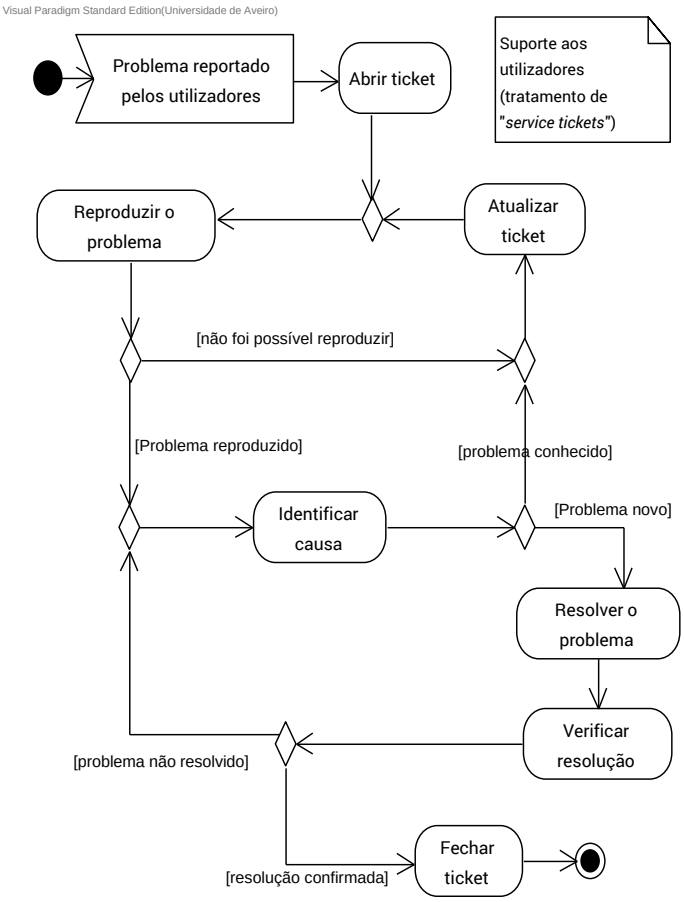

*Diagrama 2- Processo de trabalho associado a um serviço de apoio ao cliente (*service desk*), que recebe e analisa problemas (designados* tickets*).*

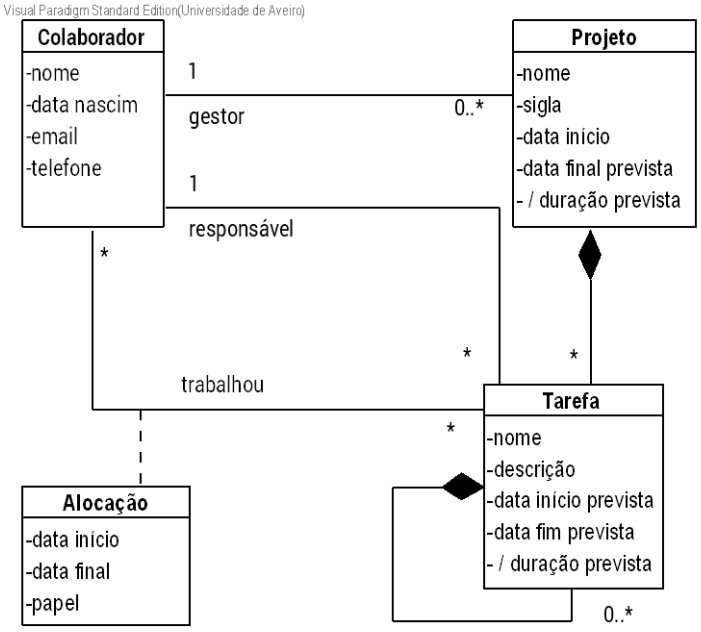

Diagrama 3 – Representação parcial dos conceitos associadas à gestão de projetos.