deti universidade de aveiro departamento de eletrónica, telecomunicações e informática

Exame de Época de Recurso – 2017-07-04. 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 que possam ser consideradas verdadeiras, pretende-se a mais abrangente); as **respostas erradas descontam** ½ da cotação; as respostas assinaladas de forma ambígua 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)																				

B1.

Qual das seguintes vantagens **não é correto** associar à UML:

a) Promove a comunicação mais clara das especificações de um sistema, numa equipa.
b) Permite diferentes níveis de análise, sendo possível ajustar o nível de detalhe dos modelos às necessidades da equipa/projeto.
c) Fornece uma sequência clara de passos para o desenvolvimento de um sistema.
d) Suporta, em parte, a geração automática de código.
e) É uma norma internacional na especificação de sistemas de software.

B2.

Quais as atividades associadas à etapa de Análise do SDLC:

a) Caraterização do valor do sistema para a organização, definição do âmbito, construção de um plano de trabalho.
b) Desenvolvimento de protótipos exploratórios, iterações iniciais sobre o modelo da base de dados.
c) Investigação do contexto existente, levantamento de requisitos, proposta de um conceito da solução.
d) Definição da arquitetura do sistema, elaboração do modelo da base de dados, desenho de interfaces com o utilizador.
e) Estudo de viabilidade, protótipos iterativos, práticas de integração contínua.

B3.

Considere o requisito a seguir formulado relativo à operação de uma loja *online* de instrumentos musicais:

R_i: "O website da loja deve ser fácil e agradável de utilizar*"*.

a) É um exemplo de um atributo de qualidade, relacionado com a facilidade de uso do sistema (*usability*).
b) É um exemplo de um requisito não funcional, relacionado com a fiabilidade do sistema (*reliability*).
c) É um exemplo de um requisito não funcional, relacionado com a compatibilidade do sistema (*suportability*).
d) Não é um bom exemplo de um requisito, porque não é objetivo e verificável.
e) Não é um bom exemplo de um requisito porque não tem a estrutura de caso de utilização.

B4.

O método OpenUP recomenda a organização do projeto em quatro fases, que terminam com pontos de verificação (*milestones)*, por esta ordem:

a) Decisão de avançar ou não com o projeto tomada; arquitetura validada; produto implementado; produto instalado no ambiente alvo.
b) Visão do projeto partilhada; casos de utilização priorizados; desenvolvimento interativo; produto instalado e documentação produzida.
c) *Backlog* definido pelo utilizador; protótipo exploratório validado; priorização dos casos de utilização; solução implementada.
d) Conceção do produto, elaboração dos requisitos, construção de incrementos, transição iterativa.
e) Requisitos definidos à cabeça, especificações confirmadas por ambas as partes, produto implementado, produto em testes.

B5.

O mesmo tipo de diagrama da UML pode ser usado para criar modelos com diferentes perspetivas de análise, em diferentes fases do SDLC, como por exemplo o ___, usado na fase de Análise para representar ___ e na fase de Construção para representar ___.

a) Diagrama de sequência/processos de trabalho da organização/ os casos de utilização.
b) Diagrama de classes/ processos de trabalho da organização / classes de código.
c) Diagrama de objetos/ conceitos do domínio/ métodos de cada classe.
d) Diagramas de atividades/ processos de trabalho da organização / algoritmos.
e) Diagrama de casos de utilização/ atores / os "logins" do sistema.

B6.

Os casos de utilização são usados como a forma principal de agrupar os requisitos funcionais (requisitos em contexto).

a) Isto é limitativo, porque apenas capta funcionalidades de que os utilizadores se lembram na altura da especificação.
b) Isto leva a especificações incompletas, porque não é possível identificar requisitos não funcionais.
c) Isto é limitativo, porque só utiliza um tipo de diagrama da UML.
d) Isto ajuda a evitar a sobre-especificação de requisitos desnecessários, mantendo a especificação em ligação com os objetivos dos utilizadores.
e) Esta forma de pensar nos requisitos funcionais deve é a principal diferença dos métodos ágeis e as abordagens sequenciais.

B7.

No cerne do TDD está um ciclo de pequenos passos que se repete, nesta ordem:

a) Remover duplicados; melhorar a clareza do código; implementar um pequeno incremento; escrever um teste para confirmar que o incremento faz o esperado.
b) Adicionar um pequeno teste; executar os testes e verificar que o novo está a falhar; implementar as alterações suficientes para o teste passar; executar os testes e verificar que o novo passa; ajeitar o código para o tornar mais limpo.
c) Ajeitar o código para o tornar mais limpo; adicionar um teste detalhado; implementar as alterações necessárias para o teste passar; executar os testes e verificar que o novo passa.
d) Implementar as alterações relativas ao novo incremento; adicionar um teste; executar os testes até se verificar que o novo teste passa.
e) Executar o código; identificar prolemas com o incremento implementado; usar ferramenta de *debugging* para encontrar a origem; resolver os erros.

B8.

A arquitetura trata da tomada das grandes decisões técnicas em relação ao sistema a desenvolver. Um exemplo de uma decisão de arquitetura é:

a) O desenho de classes de código que maximizam a coesão e minimizam a interdependência.
b) A distribuição de atributos pelas classes.
c) A distribuição de métodos pelas classes.
d) Os casos de utilização que são incluídos no âmbito do projeto.
e) A tipologia de plataformas de utilização que devem ser suportadas (*web*, Android, etc.).

B9.

Na abordagem orientada por objetos, uma classe define um "molde" do qual se pode obter objetos concretos, com estado interno e _____ que atuam sobre esse estado interno.

a) Instâncias públicas.
b) Componentes.
c) Métodos.
d) Atributos.
e) Pacotes.

B10.

Um ponto importante na modelação de sistemas é mostrar como é que os objetos do domínio de um problema irão interagir em conjunto para formar uma colaboração que realiza _____.

a) Os cenários dos casos de utilização.
b) As classes da etapa da análise.
c) Os conceitos do modelo do domínio.
d) A troca de mensagens entre objetos.
e) Todas as opções anteriores estão corretas.

B11.

Considere que se pretende modelar o protocolo de interação entre um dispositivo médico e um *smartphone*, que consegue obter valores do primeiro, usando um canal de comunicação sem fios. O que recomenda usar?

a) O Diagrama de Casos de Utilização, para identificar os usos possíveis do sistema.
b) Um diagrama estrutural, como diagrama de classes, para representar as entidades evolvidas e os seus atributos.
c) Um diagrama de interação, especialmente um que evidencie a linha temporal.
d) Um diagrama de estados, para clarificar os estados possíveis dos dispositivos que colaboram.
e) Um Diagrama de Sequência de Sistema, para captar as operações de sistema implicadas na colaboração.

B12.

Quais os principais diagramas da UML que um programador pode usar para visualizar a estrutura e comportamento do código em Java:

a) Diagrama de objetos, diagrama de atividades.
b) Diagrama de componentes, diagrama de classes.
c) Diagrama de pacotes, diagrama de componentes.
d) Diagrama de classes, diagrama de componentes.
e) Diagrama de classes, diagrama de sequência.

B13.

Relativamente ao Diagrama 1, qual o problema com os atores modelados?

a) A Base de Dados é parte do sistema sob especificação, logo não é um ator.
b) A Base de Dados é um sistema, logo não é um ator.
c) O Gestor dos espaços deve especializar o Utente, pois também pode beneficiar do aluguer de espaços.
d) Faltam associações entre a Base de Dados e todos os casos de utilização (de alguma forma leem ou escrevem na BD).
e) Não há problemas: os atores são adequados.

B14.

No Diagrama 1, as duas situações de <<include>> modeladas:

a) Refletem a dependência temporal dos casos de utilização (e.g.: Procurar não pode ser feito antes do Adicionar)
b) Estão mal aplicadas: o caso de utilização "incluído" não deve ter associação direta com atores.
c) Servem para evidenciar a necessidade da intervenção dos vários atores naqueles cenários.
d) Estão incompletas: faltam os respetivos pontos de extensão ("*extension points*").
e) Estão mal aplicadas: os cenários em causa são independentes entre si.

B15.

Como é que diagramas do género do Diagrama 1 são utilizados ao longo do SDLC?

a) Podem ser usados para detalhar/suplementar os conceitos identificados no modelo do domínio.
b) Podem ser detalhados/suplementados com diagramas de sequência
c) Podem ser substituídos por diagramas de atividades, em que há partições correspondentes aos atores.
d) Os métodos ágeis de desenvolvimento privilegiam a comunicação sobre a documentação e não recomendam o uso dos casos de utilização.
e) São usados apenas na fase de análise do sistema, para explorar requisitos funcionais.

B16.

Considere o modelo do Diagrama 2:

a) Um Equipamento pode ser usado em vários Complexos desportivos.

b) Só as Entidades de um Município podem operar/gerir Piscinas.
c) Um Utente pode, numa Reserva, incluir vários Equipamentos.
d) Uma Instalação destina-se à prática de uma Modalidade desportiva.
e) A Disponibilidade semanal (períodos de funcionamento) de um Equipamento é igual ao longo das várias semanas.

B17.

Segundo o Diagrama 2, o que é que um Utente reserva?

a) A utilização de um Complexo Desportivo, por um período de tempo.
b) A utilização de Uma Piscina municipal, por um período de tempo.
c) A utilização de um Equipamento desportivo, para realizar uma Modalidade bem definida.
d) A utilização de um Equipamento desportivo, por um período de tempo, sem especificar a Modalidade.
e) A utilização de um Equipamento desportivo, com um custo dependente da Modalidade desportiva.

B18.

Que alterações ao Diagrama 2 seriam necessárias para que o modelo tivesse a capacidade expressiva para representar o requisito "O custo hora de um Equipamento depende da Modalidade que o vai usar"?

a) Nenhuma (já é possível representar corretamente essa informação).
b) O atributo custo, que existe em Equipamento, deve ser movido para a classe Modalidade.
c) O atributo custo deve ser movido para uma classe de associação (entre Equipamento e Modalidade).
d) Deve ser acrescentado um atributo em Equipamento para representar a Modalidade.
e) O atributo custo em Equipamento pode ser descartado, porque é redundante com o que existe em Reserva de espaço.

B19.

O Diagrama 3 representa o fluxo de trabalho orientado por Histórias ("user stories"), em que:

a) A criação de Histórias leva à atualização do *Backlog*.
b) A aplicação dos controlos automáticos de garantia de qualidade não aborta o circuito de tratamento da História.
c) O Dono do produto pode rejeitar uma História, que retorna ao Programador.
d) O circuito de tratamento da História não é cancelado por eventos externos.
e) Todas as hipóteses anteriores estão corretas.

B20.

Em que ponto(s) da atividade modelada no Diagrama 3 seria de esperar que fosse usado, como *input*, os resultados do Diagrama 1?

a) Escrever a "user story".
b) Entregar a implementação.
c) Aceitar a "user story".
d) Todas as alíneas anteriores.
e) Na verdade, a atividade modelada no Diagrama 3 não usa as entidades modeladas no Diagrama 1.

B21. [questão de desenvolvimento]

"As empresas agora operam num ambiente que está a mudar de forma incrivelmente rápida. Novos produtos aparecem e desaparecem, as leis e regulamentos mudam, as empresas fundem-se e reestruturam-se. Os novos pacotes de software têm de ser rapidamente concebidos, implementados e entregues. Não há tempo para processos de engenharia de requisitos prolongados. O desenvolvimento começa logo que uma visão para o software está disponível; os requisitos emergem e são clarificados durante o processo de desenvolvimento." In: Sommerville, I. (2005). Integrated requirements engineering: A tutorial. *IEEE software*, *22*(1), 16-23.

A citação de alguma forma passa a ideia que não se pode ter agora o mesmo processo de construção de software que se teve antes. Porquê? O que pode ser feito para reduzir o elevado risco que decorre do contexto apresentado (especialmente em relação à engenharia de requisitos)?

B22. [questão de desenvolvimento]

Considere o trecho de código seguinte, em Java, que representa uma implementação incompleta associada a um serviço de gestão de reservas de equipamentos desportivos. A partir da informação que se pode inferir do trecho de código:

a) Apresente um diagrama de sequência para representar a interação que ocorre quando é invocado o método GstorDeReservas#**permuitarReservaEntreUtentes()**.
b) Apresente uma visualização da estrutura das classes. Considere apenas a classe Reserva e as classes para as quais ela tem alguma forma de dependência direta (*coupling*).

```
public class GestorDeReservas {
    private RepositorioReservas colecaoReservas;
[…]
    //Troca o titular de uma reserva com o de outra, dados os códigos de duas reservas
    public void permutarReservaEntreUtentes(int codigo1, int codigo2) {
        Reserva reserva1 = colecaoReservas.procurarReservaPorCodigo( codigo1 );
        Reserva reserva2 = colecaoReservas.procurarReservaPorCodigo( codigo2 );

        if( reserva1 != null && reserva2 != null) {
            Utente utente = reserva1.getDetalhesUtente();
            reserva1.setUtente( reserva2.getDetalhesUtente() );
            reserva2.setUtente( utente);
        }
    }
}
```

FOLHA DE DIAGRAMAS

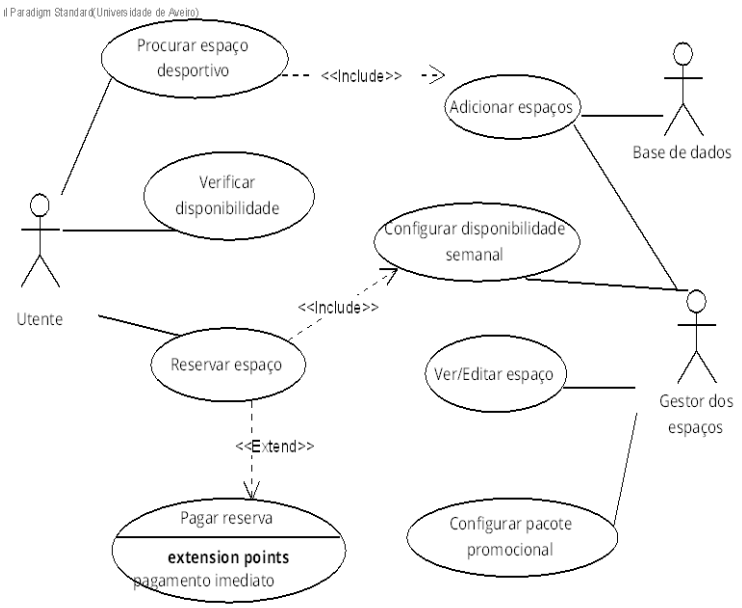

Diagrama 1- Cenários de utilização associados a um sistema de reserva de espaços desportivos.

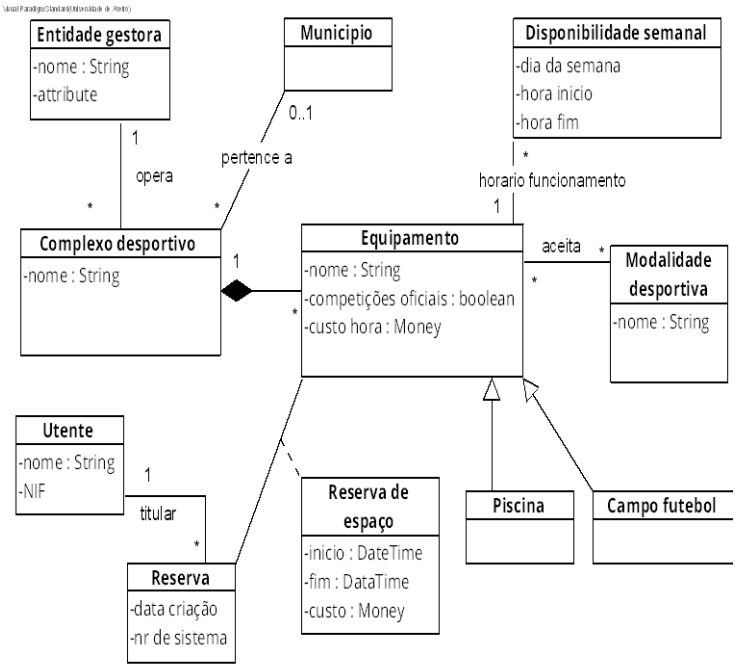

Diagrama 2 – Representação parcial dos conceitos associadas à gestão de alugueres de espaços desportivos.

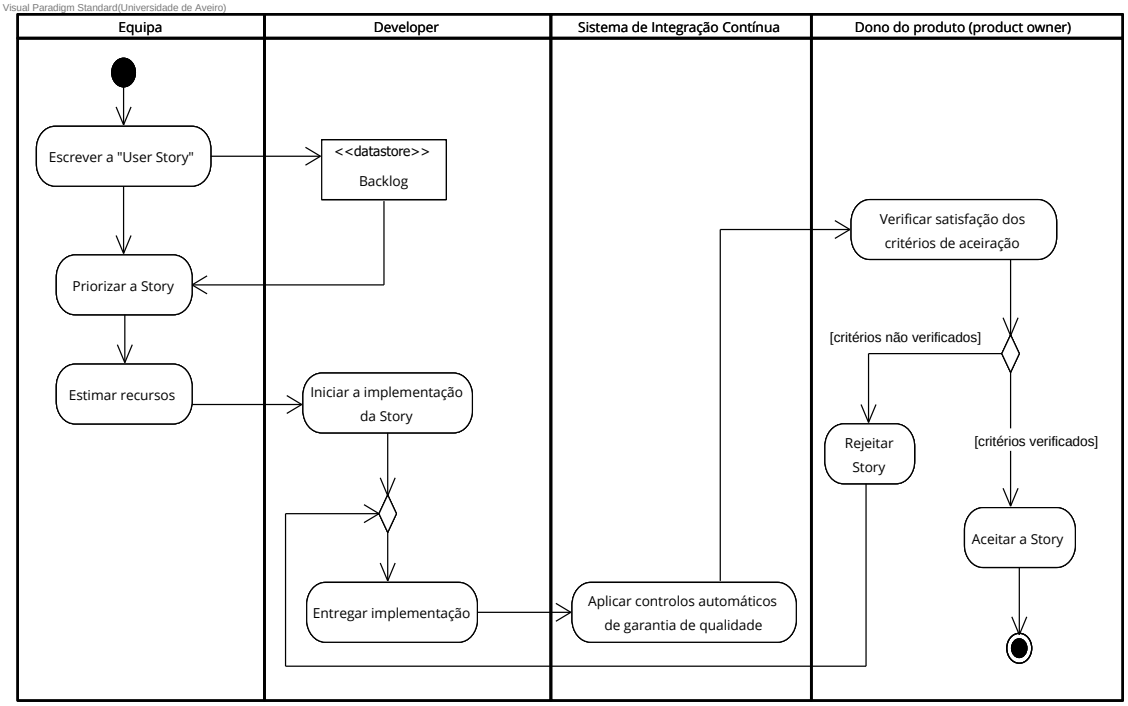

Diagrama 3- Processo de trabalho numa equipa de desenvolvimento.