Fundação CECIERJ – Vice-Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Engenharia de Software
AP2 2° semestre de 2018.

Nome –

Assinatura –

Observações:

1. Prova sem consulta, sem uso de máquina de calcular. E sem celular!
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

---

1. Pergunta da AD2: Apresente duas similaridades e duas diferenças entre a engenharia de sistemas de software contemporâneos (IoT, Industria 4.0, Sistemas de Sistemas, Sistemas Sensíveis ao contexto, dentre outros) e a engenharia de sistemas de software convencionais (web, cliente/servidor, dentre outros). Indique um desafio associado a engenharia do software contemporâneo. (valor: 2,0 pontos).

Os chamados sistemas contemporâneos são essencialmente diferentes em suas propriedades. Esses sistemas deixaram de ser formados por blocos monolíticos sendo executados em um único ou poucos dispositivos e passaram a ser compostos por diferentes "coisas" que possuem sua própria inteligência e comportamento.

Esses novos sistemas requerem diversas características não comumente encontradas nos sistemas de software tradicionais. Interoperabilidade intensa, descoberta de serviços, escalabilidade, disponibilidade, privacidade e segurança são algumas delas.

Dessa forma, esse novo tipo de sistema requer uma nova abordagem de engenharia que possa contemplar a construção de sistemas enfatizando essas novas propriedades.

Referências:

Atzori, L, Iera, A and Morabito, G. "The Internet of Things: A Survey" The International Journal of Computer and Telecommunications Networking (2010)

Belkeziz, R and Jarir, Z. “A Survey on Internet of Things Coordination” (2016)

Top 50 IoT projets. Disponível em http://www.libelium.com/resources/top_50_iot_sensor_applications_ranking/

Design Principles for Industrie 4.0 Scenarios, Hawaii International Conference on System Sciences - Industrie 4.0: Hit or Hype?

2. Explique a relação que existe entre as atividades de gerenciamento de tempo e de gerenciamento de custos em projetos de desenvolvimento de software. (2,0 pontos).

No gerenciamento de tempo definimos um cronograma em cujas atividades será realizado todo o esforço compreendido no projeto. Estas atividades consumirão recursos para que o trabalho nelas compreendido possa ser realizado. Estes recursos têm custo e o custo do projeto poderá ser calculado a partir dos custos dos recursos utilizados nestas atividades. Sendo assim, o cronograma desenvolvido no gerenciamento de tempo é o fator gerador dos custos que serão administrados no gerenciamento de custos.

3. Para as métricas a-e abaixo, indique seu nível de precisão (objetiva, subjetiva), escala (nominal, ordinal, intervalar, razão) e objeto de medição (processo, produto). Para cada uma delas, apresente 1 (um) exemplo de seu uso no contexto dos projetos de software, comentando sobre o comportamento ou fenômeno do software associado a medida da métrica. (Valor: 2,0 pontos; máximo 10 linhas)

    a. Número de Defeitos
    b. Complexidade Ciclomática
    c. Perda de Coesão em Métodos
    d. Acoplamento entre Objetos
    e. Usabilidade

a. Número de defeitos: objetiva, razão, produto. Usada para medir a qualidade do software sob o ponto de vista dos seus usuários.

b. Complexidade Ciclomática: Objetiva, razão, produto. Usada para medir o número de ciclos fechados no grafo de programa. Associada ao número de pontos de decisão do algoritmo e a indicação de “esforço de teste” do algoritmo.

c. Perda de Coesão em Métodos: Objetiva, razão, produto. Indica o grau de coesão dos componentes de uma classe.

d. Acoplamento entre Objetos: Objetiva, razão, produto. Usada para medir o nível de dependência entre classes/objetos.

e. Usabilidade: subjetiva, nominal, produto. Usada para indicar a facilidade de uso de um software.

4. Defina ***falta***, ***erro***, ***falha*** e ***defeito*** no contexto de testes de software. (valor 2 pontos).

Falta: a manifestação de um erro no software

Erro: ação humana que produz um resultado incorreto.

Falha: Evento notável onde o sistema viola suas especificações.

Defeito: Deficiência mecânica ou algorítmica que se ativada pode levar a uma falha. Podem ser classificados como omissão, inconsistência, fato incorreto, ambiguidade e informação estranha.

Testes de software revelam falhas. As falhas são consequência dos defeitos (faltas provocadas por erros dos desenvolvedores).

5. Alguns critérios podem ser utilizados para projetar casos de teste. Por exemplo, análise do valor limite é um deles, usado para teste funcional. Dê outro exemplo de critério, indicando se ele deve ser usado para teste funcional ou teste estrutural e o aplique no requisito abaixo para explicar sua utilização e gerar os casos de teste (Valor 2 pontos):

Em uma indústria química o controle da temperatura e pressão de um determinado processo depende de dois fatores: a temperatura de ebulição do produto sendo processado e de um coeficiente Z. As entradas do programa de controle são a temperatura de ebulição do produto em graus Celsius limitada ao intervalo [90,200] e o coeficiente Z limitado ao intervalo [0.3, 2.0]. O sistema de controle deve suspender o processo se a temperatura ou a pressão atingirem valores críticos. O cálculo do valor crítico da temperatura é obtido pela fórmula: tk = te*2. O valor crítico da pressão pela fórmula:

pk = tk * Z, quando te for menor que 100;

pk = te * Z nos demais casos.

Onde, tk=temperatura crítica, te=temperatura de ebulição, pk=pressão crítica.

Como este software foi construído visando *testabilidade*, é possível entrar com um terceiro e quarto valores correspondentes à temperatura e a pressão em um dado instante. Nesse caso, o software efetua os cálculos e imprime uma mensagem sinalizando se interromperia ou não o processo.

Cálculo dos parâmetros de teste:

1) Temperatura Crítica: 400ºC (200 x 2)

2) Pressão Crítica: Te <100: [54...400] e Te >100: [30...400]

Desta forma teríamos as seguintes situações para o teste:

Parâmetro	Inválida	Pk (te < 100)	Pk (te >100)	Inválida
te	<90	te * 2 * Z	te *Z	> 200

Observe que a combinação de temperatura e pressão compõe o requisito. Assim, as classes válidas e inválidas devem ser entrelaçadas já que o processo deve ser interrompido em qualquer situação que temperatura ou pressão assumam valores críticos. Como estamos trabalhando com valores críticos (muito baixos ou muito altos), Z deve variar de 0.3 a 2.0 (mínimo, máximo) para a definir a faixa aceitável de pressão crítica.

Precisamos de um caso de teste para cada classe inválida (<90, >200) e independentemente da pressão, o processo dever ser interrompido: {85, --, interromper processo}; {201, --, interromper processo}.

Precisamos de três casos de teste para o intervalo [90,100]: {95, 100, manter processo}; {95, 52, interromper processo}; {95, 401, interromper processo}.

Precisamos de três casos de teste para o intervalo [100, 200]: {150, 300, manter processo}; {150, 25; interromper processo}; {150, 401, interromper processo}.

Boa Prova!