PLANO DE ENSINO ENGENHARIA DE SOFTWARE **Implantação 20182**

CARGA HORÁRIA: 66h Teórica: 33h Prática: 33h

EMENTA

Apresentar os conceitos de engenharia de software, os processos de software e produtos de software. Abordar os ciclos de vida de sistemas e seus paradigmas, engenharia de requisitos, validação, verificação e teste de software, além de manutenção e evolução de software. Enfoca projeto de software orientado a objetos, com diagramas UML. Gerência e Configuração de Mudanças

COMPETÊNCIAS

II – TRABALHAR EM EQUIPE
III – ATINGIR OBJETIVOS
VII - CULTURA DIGITAL

IX. PENSAMENTO LÓGICO - Pensar e usar a lógica formal estabelecendo relações, comparações e distinções em diferentes situações.
X. REPRESENTAÇÃO ESPACIAL E MODELAGEM - Representar graficamente desenhos manuais e modelos, através das técnicas apropriadas.
XIII. ADMINISTRAÇÃO E GERENCIAMENTO - Gerenciar recursos, tempo e processos visando a tomada de decisão e a otimização dos resultados.
XIV DOMÍNIO DA TECNOLOGIA DA INFORMAÇÃO E COMUNICAÇÃO - Utilizar sistemas informatizados requeridos para a operacionalização da profissão.
XV. VISÃO ESTRATÉGICA - Planejar ações a curto, médio e longo prazo para atingir metas, antecipando tendências e novas oportunidades.

XVII - CRIAÇÃO E DESENVOLVIMENTO DE SISTEMAS - Projetar, desenvolver e implementar sistemas computacionais objetivando a integração de recursos físicos e lógicos. (ciência dos dados)
XVIII - CRIAÇÃO E DESENVOLVIMENTO DE HARDWARE - Projetar, desenvolver e implementar dispositivos eletrônicos e microprocessados. (ciência dos dados)
XX - GERENCIAMENTO E EXECUÇÃO DE PROJETOS - Gerenciar, executar e realizar manutenção de projetos de sistemas, hardware e de soluções para automação. (ciência dos dados)
XIX - REQUISITOS DE SOFTWARE E PROJETO DE INTERFACE - Especificar e gerenciar requisitos de software, gerenciar configurações de projeto de software e o projeto de interfaces. (análise e desenvolvimento de sistemas)
XX - MANUTENÇÃO DE SISTEMAS - Realizar rotinas demandadas e programadas de verificação nos sistemas de informação implementados. (análise e desenvolvimento de sistemas)
XX - GESTÃO DE PROJETOS EM TI - Aplicar conceitos, métodos, técnicas e ferramentas de gerenciamento de projetos em sua área de atuação. (gestão da tecnologia da informação)
XXI - TECNOLOGIA DA INFORMAÇÃO - Identificar oportunidades de mudanças e projetar soluções usando tecnologias da informação nas organizações. (gestão da tecnologia da informação)
XXI - GESTÃO DE PROJETOS DE BIG DATA - Gerir projetos de implantação de ferramentas de Big Data e Inteligência Analítica em organizações. (big data e inteligência analítica)
XXII – GESTÃO DE PROJETOS - Gerenciar projetos de desenvolvimento de sistemas computacionais. (ciência da computação)
XXII - TECNOLOGIA DA INFORMAÇÃO - Identificar oportunidades de mudanças e projetar soluções usando tecnologias da informação nas organizações. (sistemas de informação)

OBJETIVOS DE APRENDIZAGEM

- Descrever o cenário atual da indústria de software
- Compreender o processo de desenvolvimento de software
- Identificar os princípios necessários e das qualidades desejadas no processo de desenvolvimento de software.
- Analisar os aspectos envolvidos na engenharia de requisitos, no gerenciamento e na qualidade de projetos de software.
- Analisar os modelos de processo de desenvolvimento de software
- Desenvolver produtos de software
- Avaliar processos de desenvolvimentos de software, tais como modelos tradicionais e modelos de desenvolvimento ágil e avaliar sua aplicabilidade no contexto de negócio.
- Identificar, analisar e documentar requisitos e regras de negócio de um software.
- Identificar os vários modelos de ciclo de vida e seu efeito na prática da produção de software.
- Conhecer e saber aplicar métodos e ferramentas de especificação de sistemas de informação.
- Conhecer os conceitos de projeto de sistemas de informação e capacitar-se na utilização de seus métodos, técnicas e ferramentas.
- Identificar as etapas de implementação, teste e manutenção de sistemas de computação e ser capaz de realizá-los e/ou coordená-los.
- Conhecer e saber aplicar métodos de controle da qualidade do processo de software.

ATIVIDADE PRÁTICA SUPERVISIONADA

Objetivos	Atividades a serem desenvolvidas	Avaliação
1. Analisar as características, vantagens e desvantagens dos processos de desenvolvimento de software; 2. Comparar e Analisar as características dos modelos de desenvolvimento tradicionais e Ágeis; 3. Avaliar a aplicação do modelo Cascata, espiral, RUP, o Scrum e o Lean no desenvolvimento de software. 4. Analisar o manifesto ágil no contexto de desenvolvimento de software. 5. Aplicar Scrum no desenvolvimento de um software.	As atividades que serão desenvolvidas pelos estudantes encontram-se detalhadas no ambiente virtual de aprendizagem (Blackboard) da disciplina.	A avaliação das APS será baseada em um padrão de correção conhecido como rubrica, que confere transparência às expectativas em relação à performance do estudante. São esses padrões que o professor utilizará ao corrigir sua APS (peso 1) que, é um dos instrumentos avaliativos.

EDITAL DE PRÁTICA DA DISCIPLINA: Baixe aqui

CRONOGRAMA DE AULAS

1. INTRODUÇÃO A ENGENHARIA DE SOFTWARE
- Desenvolvimento do profissional de software.
- Ética na engenharia de software.

Objetivos de Aprendizagem	Competências Relacionadas
1. Introduzir, Conceituar engenharia de software e identificar a sua importância. 2. Apresentar os diferentes papéis na Engenharia de Software e profissionais envolvidos. 3. Relacionar o desenvolvimento de softwares distintos a técnicas diferentes de engenharia de software. 4. Abordar as Ferramentas CASE (*Computer-Aided Software Engineering* - Engenharia de Software Auxiliada por Computador) e seus tipos. 5. Analisar algumas questões éticas e profissionais para engenheiros de software.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Pesquisa diagnóstica / Aula expositiva interativa / Brainstorm Sequência sugerida: • Apresentação dos objetivos de aprendizagem, sistema de avaliação e principais referências; • Levantamento dos conhecimentos prévios por meio de Brainstorm; • Aula expositiva interativa: apresentação dos conceitos em engenharia de software, como desenvolve-se um engenheiro de software e o código de ética em engenharia de software. • Atividade prática: ler e resumir o código de ética dos engenheiros de software. • Cinco perguntas sobre o assunto no Kahoot ou Socrative. Caso não haja acesso à Internet, o questionário pode ser aplicado na forma impressa ou nos slides finais da aula. • Atividade de fixação individual extraclasse.	Quiz rápido apresentando as questões no ppt. Socrative ou Kahoot Minute paper: escreva um parágrafo refletindo sobre o processo de aprendizagem ocorrido na aula.	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 01, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 1. Pearson Addison Wesley. 2011. Pfleeger, S. L. Engenharia de Software - Teoria e Prática Capítulo 01 - Prentice Hall PADUA F, Paula W. Engenharia de Software Capítulo 01 - Fundamentos, Métodos e Padrões, 3ª edição. LTC, 11/2008.

2. CICLO DE VIDA E MODELOS DE DESENVOLVIMENTO DE SOFTWARE
Cascata, Espiral, Prototipação, Modelo em V, RAD e RUP
- Ciclo de Vida Evolutivo, Iterativo e Incremental.

Objetivos de Aprendizagem	Competências Relacionadas
1. Definir Ciclo de Vida no Desenvolvimento de Software 2. Conceituar os Ciclos de Vida Evolutivo, Iterativo e Incremental 3. Descrever os modelos de processos de software. 4. Identificar e Utilizar modelos de processos de software. 5. Identificar por que os processos devem ser organizados de maneira a lidar com as mudanças nos requisitos e projeto de software. 6. Identificar como o RUP integra boas práticas de engenharia de software na criação de processos de software adaptáveis.	I II XII XIV

	Estratégias de Ensino	Avaliação Formativa	Recursos
	Perguntas e respostas / Aula expositiva interativa / Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: perguntas e respostas • Aula expositiva dialogada: apresentação dos modelos de processo de software, das atividades do processo. • Atividade prática: analisar a aplicação d RUP em um estudo de caso fornecido pelo professor. • Cinco perguntas sobre o assunto no Kahoot ou Socrative. Caso não haja acesso à Internet, o questionário pode ser aplicado na forma impressa ou nos slides finais da aula. • Atividade de fixação individual extraclasse	Os alunos devem se agrupar em, para discorrer, dialogar e apresentar um relatório com os principais modelos de desenvolvimento tradicionais. Questionário sobre o assunto no final da aula usando os últimos slides, Kahoot ou Socrative.	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 02, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 2. Pearson Addison Wesley. 2011. Pfleeger, S. L. Engenharia de Software - Teoria e Prática Capítulo 01 - Prentice Hall PADUA F, Paula W. Engenharia de Software Capítulo 01 - Fundamentos, Métodos e Padrões, 3ª edição. LTC, 11/2008.

3. CICLO DE VIDA E MODELOS DE DESENVOLVIMENTO DE SOFTWARE - RUP - Processo Unificado da Rational - IBM	Objetivos de Aprendizagem	Competências Relacionadas
	1. Definir Ciclo de Vida no Desenvolvimento de Software 2. Conceituar os Ciclos de Vida Evolutivo, Iterativo e Incremental 3. Descrever os modelos de processos de software. 4. Identificar e Utilizar modelos de processos de software. 5. Identificar por que os processos devem ser organizados de maneira a lidar com as mudanças nos requisitos e projeto de software. 6. Identificar como o RUP integra boas práticas de engenharia de software na criação de processos de software adaptáveis.	I II XII XIV

	Estratégias de Ensino	Avaliação Formativa	Recursos
	Perguntas e respostas / Aula expositiva interativa / Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: perguntas e respostas • Aula expositiva dialogada: apresentação dos modelos de processo de software, das atividades do processo. • Apresentar Cases exemplos e documentos que apliquem os modelos de desenvolvimento tradicionais (mostrar um documento de software exemplo); • Atividade prática: aplicar RUP em um estudo de caso fornecido pelo professor. • Cinco perguntas sobre o assunto no Kahoot ou Socrative. Caso não haja acesso à Internet, o questionário pode ser aplicado na forma impressa ou nos slides finais da aula. • Atividade de fixação individual extraclasse	Os alunos devem se agrupar em, para analisar um case que utilize o RUP aplicado ao desenvolvimento de um software. Questionário sobre o assunto no final da aula usando os últimos slides, Kahoot ou Socrative.	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 02, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 2. Pearson Addison Wesley. 2011. Pfleeger, S. L. Engenharia de Software - Teoria e Prática Capítulo 01 - Prentice Hall PADUA F, Paula W. Engenharia de Software Capítulo 01 - Fundamentos, Métodos e Padrões, 3ª edição. LTC, 11/2008.

4. MÉTODOS ÁGEIS DE DESENVOLVIMENTO DE SOFTWARE
- Manifesto Ágil, Os doze (12) princípios do manifesto
- Principais modelos: ASD, DSDM, FDD, LSD (Lean Software Development), AM, AUP, Crystal, Kanban, SCRUM e XP.
Práticas Ágeis

Objetivos de Aprendizagem	Competências Relacionadas
1. Descrever os métodos ágeis de desenvolvimento de software. 2. Descrever o manifesto ágil. 3. Diferenciar desenvolvimento ágil de desenvolvimento tradicional. 4. Identificar as práticas Agile. 5. Usar a abordagem Agile. 6. Discutir questões de escalonamento de métodos ágeis.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Mapa conceitual / Aula expositiva interativa / Estudo de caso / Exemplo de Aplicação em Projetos Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: mapa conceitual; • Aula expositiva dialogada: Contexto histórico, Manifesto Ágil, Os doze princípios do manifesto ágil, AGILE, os métodos e práticas ágeis de desenvolvimento de software. • Aplicação de minute paper • Atividade de fixação individual extraclasse	Questionários Minute paper: Os alunos devem estar agrupados e fazer um resumo sobre o assunto da aula. Exercícios de Fixação	SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 3. Pearson Addison Wesley. 2011. PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 05, 8th edição. AMGH, 01/2016 www.desenvolvimentoagil.com.br www.manifestoagil.com.br/ metodologiaagil.com/ www.culturaagil.com.br/o-que-sao-metodos-ageis/

5. MÉTODOS ÁGEIS DE DESENVOLVIMENTO DE SOFTWARE
SCRUM – Gerência Ágil de Projetos e KANBAN
XP – Extreming Programming – Programação Extrema
LEAN – LSD Lean Software Development - Desenvolvimento Enxuto

Objetivos de Aprendizagem	Competências Relacionadas
1. Conhecer os frameworks ágeis SCRUM e XP. 2. Identificar as características de cada FRAMEWORK ágil e descrê-las. 3. Diferenciar Metodologia e Framework 4. Vivenciar, por meio de dinâmicas, o planejamento de um projeto ágil, utilizando o SCRUM. 5. Desenvolver habilidades de trabalho em equipes	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Aula expositiva dialogada / Estudo de caso / Dinâmica de Estimativas Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: revisão e atividades sobre a aula anterior. • Aula expositiva dialogada: apresentação do SCRUM e KANBAN, práticas, eventos e cerimônias. Os papéis: Product Owner, Scrum Master e Time de Desenvolvimento. Artefatos: Product Backlog, Sprint Backlog, Burndowchart e Histórias de Usuário. • Aula expositiva dialogada: apresentação do XP, valores, princípios, papéis e práticas. • Atividade prática: análise de um caso e confecção dos artefatos: Product Backlog, Sprint Backlog, Cartão de Histórias de Usuário. Jogo de estimativas de tarefas utilizando o planning poker. Quadro de Tarefas com Kanban • LEAN • Atividade de fixação individual extraclasse	Parte 01: Alunos formam grupos (4 pessoas – Product Owner, Scrum Master e Time de Desenvolvimento), dividem - se entre os papéis do Scrum, pensam as tarefas do projeto e produzem o Product Backlog planejando as tarefas do projeto (com cronograma). Dividem as tarefas entre as Sprints, jogam planning poker calculando as estimativas das tarefas e por fim elaboram para cada tarefa as histórias de usuário. Grupos organizam-se e planejam o desenvolvimento do projeto dividindo as responsabilidades. Avaliação do Trabalho em EQUIPE. Parte 02: Entrega e Apresentação do Trabalho em data posterior (Sugestão: trabalho pode ser uma ferramenta de avaliação)	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 05, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 3. Pearson Addison Wesley. 2011. Teles V. M, Extreme Programming - 2014 – NOVATEC Sutherland J. Scrum. A Arte de Fazer o Dobro do Trabalho na Metade do Tempo – LEYA Beck K. Programação Extrema. XP Explicada - BOOKMAN- 2004 GUIA DO SCRUM BR https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf CONHECIMENTO EM SCRUM (SBOK)

			https://www.scrumstudy.com/SBOK/SCRUMstudy-SBOK-Guide-2016-Portuguese.pdf

6. ENGENHARIA DE REQUISITOS
- Requisitos funcionais e não funcionais.
- Documento de requisitos.
- Especificação de requisitos.
- Processos de engenharia de requisitos

Objetivos de Aprendizagem	Competências Relacionadas
1. Conceituar os requisitos de usuário e de sistema. 2. Identificar e conhecer a hierarquia e níveis de requisitos. 3. Identificar porque os requisitos de usuário e de sistema devem ser escritos de formas diferentes. 4. Distinguir requisitos funcionais, não funcionais e de domínio (regras de negócio). 5. Descrever o documento de requisitos de software.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Brainstorming / Aula expositiva interativa Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: brainstorm. • Características dos requisitos • Aula expositiva dialogada: apresentação dos conceitos de requisitos e sua hierarquia, requisitos de usuário e de sistema, funcionais, não funcionais e de domínio (regras de negócio), elaboração de um modelo de documento de requisitos, como os requisitos devem ser especificados e dos processos de engenharia de requisitos. • Leitura de artigos de Karl Wiegers: In Search of Excellent Requirements http://www.processimpact.com/articles/exc_reqs.pdf Karl Wiegers Describes 10 Requirements Traps to Avoid https://faculty.cs.byu.edu/~rodham/cs428/requirements-traps.pdf When Telepathy Won't Do: Requirements Engineering Key Practices http://ww.w.processimpact.com/articles/telepathy.pdf First Things First: Prioritizing Requirements http://www.tarrani.net/linda/prioritizing.pdf Writing Quality Requirements1 http://www.uml.org.cn/rjzl/pdf/1113/qualreqs.pdf • Atividade prática: identificar requisitos funcionais, não funcionais e regras de negócio em estudos de casos. • Autoavaliação por checklist • Atividade de fixação individual extraclasse	Divisão dos alunos em grupos colaborativos para reflexão, leitura, análise e discussão dos artigos sobre requisitos Análise de situação problema por meio de estudo de caso selecionando e classificando os requisitos. Apresentação de questões sobre o assunto no Kahoot, Socrative ou nos slides finais da aula. Exercícios de Fixação	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 08, 09, 10 e 11, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 4. Pearson Addison Wesley. 2011. Nery M. F. - Análise e Gestão de Requisitos de Software - Onde Nascem Os Sistemas - 3ª Ed. 2015 Como escrever requisitos de forma simples: https://medium.com/lfdev-blog/como-escrever-requisitos-de-software-de-forma-simples-e-garantir-o-m%C3%ADnimo-de-erros-no-sistema-app-74df2ee241cc https://www.ibm.com/developerworks/rational/library/4166.html https://www.ibm.com/support/knowledgecenter/en/SSSHCT_7.1.0/com.ibm.reqpro.help/req_concepts/c_docs.html

7. ELICITAÇÃO DE REQUISITOS
- Elicitação, validação e gerenciamento de requisitos.

Objetivos de Aprendizagem	Competências Relacionadas
1. Realizar as atividades de elicitação, análise e validação de requisitos. 2. Aplicar os conceitos de gerenciamento de requisitos. 3. Analisar e validar os requisitos de projeto e sistema 4. Realizar estudo etnográfico para detecção de requisitos essenciais 5. Identificar a importância dos Requisitos no projeto	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Brainstorming / Aula expositiva interativa Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: brainstorming, breve revisão da aula anterior. • Aula expositiva interativa: apresentação de métodos de elicitação, validação e gerenciamento de requisitos. • Análise e compreensão das diversas técnicas de elicitação de requisitos (Etnografia, Questionário, Entrevista, Brainstorm, Role Playing, JAD, Caso de Uso, Prototipação, etc). • Atividade prática: elicitação de requisitos através de casos de uso em histórias fornecidas pelo professor. Aplicação de Técnicas de elicitação de requisitos. • Aplicação de questionário curto sobre o assunto da aula. • Atividade de fixação individual extraclasse	Checklist: avaliação em pares Alunos reúnem-se em grupos, recebem os temas de projetos subdividem-se em outros dois grupos e exercem papéis de analistas e cliente, depois se alternam. Objetivo: de acordo com as características do projeto, analisar quais as técnicas são aplicáveis e quais foram adotadas pela equipe, justificando sua escolha, esta atividade gerará um relatório técnico por grupo a ser avaliado pelo professor.	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 08, 09, 10 e 11, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 4. Pearson Addison Wesley. 2011. Nery M. F. - Análise e Gestão de Requisitos de Software - Onde Nascem Os Sistemas - 3ª Ed. 2015

8. MODELAGEM DE SISTEMAS
- Modelos de contexto.
- Modelos de interação.
- Engenharia dirigida a modelos.

Objetivos de Aprendizagem	Competências Relacionadas
1. Identificar como modelos que representam sistemas de software. 2. Reconhecer por que diferentes tipos de modelos são necessários. 3. Utilizar as perspectivas fundamentais de modelagem de sistema de contexto, interação, estrutura e comportamento. 4. Descrever os modelos de contexto e de interação. 5. Aplicar os modelos de contexto e de interação.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Perguntas e respostas / Aula expositiva dialogada / estudo de caso Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: perguntas e respostas • Aula expositiva dialogada: apresentação dos conceitos de modelagem, princípios de modelagem, modelos de contexto, modelos de interação -Introdução a UML como modelo de análise orientada a objeto, diagrama de caso de uso e diagrama de sequência como exemplos de diagrama de interação. • Atividade prática: aplicar modelos de contexto e de interação num estudo de caso fornecido pelo professor. • Minute paper: Os alunos devem estar agrupados e fazer um resumo sobre o assunto. • Atividade de fixação individual extraclasse	Minute paper: Os alunos devem estar agrupados e fazer um resumo sobre o assunto. Exercícios de fixação	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 07, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 4. Pearson Addison Wesley. 2011. SCHACH, Stephen R. Engenharia de Software. Capítulo 16, ArtMed, 09/2010.

			LARMAN, Craig. Utilizando UML e Padrões. Bookman, 08/2011 FOWLER, Martin. UML Essencial. Bookman, 08/2011.

9. MODELAGEM DE SISTEMAS - Modelos estruturais. - Modelos comportamentais. - Engenharia dirigida a modelos.	Objetivos de Aprendizagem		Competências Relacionadas
	1. Descrever como modelos gráficos podem ser usados para representar sistemas de software. 2. Identificar por que diferentes tipos de modelos são necessários. 3. Aplicar as perspectivas fundamentais de modelagem de sistema de contexto, interação, estrutura e comportamento. 4. Descrever os modelos estruturais e comportamentais. 5. Aplicar os modelos estruturais e comportamentais. 6. Identificar as ideias básicas da engenharia dirigida a modelos.		I II XII XIV
	Estratégias de Ensino	Avaliação Formativa	Recursos
	Perguntas e respostas / Demonstração / Estudo de caso Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: breve revisão da aula anterior com perguntas e respostas. • Aula expositiva dialogada: apresentação dos modelos estruturais (diagrama de classes como exemplo da visão estrutural), modelos comportamentais (diagrama de atividades e estados como exemplo) e de engenharia dirigida a modelos (MDE - Model Driven Engineering, MDA – Model Driven Architeture, UML executável) . • Atividade prática: aplicar e identificar modelos estruturais e comportamentais em um estudo de caso fornecido pelo professor. • Atividade de fixação individual extraclasse	Atividade em grupo com Estudos de Caso que apliquem os modelos abordados. Atividades de fixação.	SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 4. 2011. LARMAN, Craig. Utilizando UML e Padrões. Bookman, 08/2011 FOWLER, Martin. UML Essencial. Bookman, 08/2011.

10. PROJETO DE ARQUITETURA - Decisões de projeto de arquitetura. - Visões de arquitetura. - Projeto e Implementação;	Objetivos de Aprendizagem		Competências Relacionadas
	1. Apresentar a importância do projeto de arquitetura. 2. Analisar as decisões que precisam ser tomadas sobre a arquitetura de sistema durante o processo de projeto de arquitetura. 3. Descrever os padrões de arquitetura. 4. Descrever a arquitetura em camadas; 5. Apresentar diferentes arquiteturas e exemplos		I II XII XIV
	Estratégias de Ensino	Avaliação Formativa	Recursos
	Perguntas e respostas / Demonstração / Estudo de caso Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: revisão breve da aula anterior e perguntas e respostas • Definição de Projeto e Arquitetura de Sistemas • Arquitetura de Sistemas em camadas • Importância da camada de negócio no tratamento dos dados • Demonstração: apresentação das decisões de projeto de arquitetura e das visões de arquitetura. • Atividade prática: realização de tomadas de decisões de projeto de arquitetura em um estudo de caso fornecido pelo professor.	Quiz de fixação de conceitos utilizando as ferramentas Kahoot e Socrative	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 13, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 6. Pearson Addison Wesley. 2011.

	• Os alunos devem para apresentar um relatório de modelo de contexto e modelo de interação de um software. • Atividade de fixação individual extraclasse		

11. PROJETO E IMPLEMENTAÇÃO

- Projeto orientado a objetos com UML.

Objetivos de Aprendizagem	Competências Relacionadas
1. Identificar as atividades mais importantes em um processo de projeto orientado a objetos. 2. Utilizar alguns dos diferentes modelos que podem ser usados para documentar um projeto orientado a objetos. 3. Identificar os diagramas de UML que podem ser aplicados em um projeto orientado a objetos. 4. Aplicar UML a um projeto de software.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Aula expositiva dialogada / trabalho Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Aula expositiva dialogada: apresentação dos diagramas UML para projeto de software. • O Processo de Engenharia Reversa em Modelagem de Sistemas (Ex. Diagrama de Classes gerando o escopo do código numa linguagem O.O) • Atividade prática: projeto de um software baseado em requisitos, casos de uso e diagrama de atividades. Professor apresenta e propõe aos alunos casos pequenos em sistemas reais. • Atividade de fixação individual extraclasse	Atividade em grupo – alunos aplicam a modelagem de caso de uso e diagrama de atividades num sistema real.	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 12, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9ª edição. Capítulo 7. Pearson Addison Wesley. 2011. Livro do CRAIG & LARMAN

12. PADRÕES DE PROJETOS

Objetivos de Aprendizagem	Competências Relacionadas
1. Conceituar e caracterizar padrões de projeto. 2. Identificar os principais padrões de projeto usados.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
QUIZ / Aula expositiva interativa Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: quis e breve revisão da aula anterior • Aula expositiva interativa: apresentação dos padrões de projeto de software mais utilizados, padrão GOF e outros exemplo de padrões de projeto. • Atividade prática: aplicar padrões de projetos em um exercício prático. • Aplicação de questionário curto sobre o assunto da aula. • Atividade de fixação individual extraclasse	Quiz de fixação de conteúdo utilizando ferramentas interativas: Kahoot, Socrative	A

13. TESTES DE SOFTWARE
- Testes de desenvolvimento.
- Desenvolvimento dirigido a testes.

Objetivos de Aprendizagem	Competências Relacionadas
1. Listar os estágios de teste durante o desenvolvimento. 2. Aplicar as técnicas para auxiliar na escolha de casos de teste orientados para descobrir defeitos de programas. 3. Descrever o desenvolvimento orientado a testes.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Perguntas e respostas / Demonstração / trabalho em trios / Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: perguntas e respostas • Demonstração: apresentação dos níveis, técnicas e métodos de testes e do desenvolvimento voltado a testes. • Atividade prática: aplicar os métodos de testes de desenvolvimento em pequenos softwares. • Autoavaliação por checklist • Atividade de fixação individual extraclasse	Checklist Atividade em grupo de planejamento de testes	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 22, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 8. Pearson Addison Wesley. 2011. Molinari, L. Testes de Software - Produzindo Sistemas Melhores e Mais Confiáveis, Ed. Érica

14. TESTES DE SOFTWARE
- Testes de releases.
- Testes de usuário.

Objetivos de Aprendizagem	Competências Relacionadas
1. Diferenciar testes de componentes e testes de release. 2. Descrever o funcionamento dos processos e técnicas de teste de usuário.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Aula expositiva interativa Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Aula expositiva interativa: apresentação dos métodos de testes de releases e testes de usuário. • Ferramentas de Testes automatizados • Planos de Testes, Cenários de Testes e Casos de Testes; • Atividade prática: Desenvolvimento de um plano com cenários e casos de teste de um projeto de software pequeno, um CRUD, por exemplo. • Aplicação de questionário curto sobre o assunto da aula. • Atividade de fixação individual extraclasse	Desenvolvimento de um plano de teste com cenários e casos. Kahoot ou Socrative.	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 22, 23, 24, 25 e 26 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 8. Pearson Addison Wesley. 2011. Molinari, L. Testes de Software - Produzindo Sistemas Melhores e Mais Confiáveis, Ed. Érica

15. EVOLUÇÃO DE SOFTWARE
- Processos de evolução de software.
- Dinâmica de evolução de programas.

Objetivos de Aprendizagem	Competências Relacionadas
1. Lidar com a mudança em um sistema de software. 2. Identificar o desenvolvimento e a evolução de um software 3. Descrever os processos de evolução de software.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
Aula expositiva dialogada / Mapa conceitual Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Aula expositiva dialogada: apresentação dos processos de evolução de software e das dinâmicas de evolução de programas, evolução e história.	Kahoot, Socrative	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 36 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software.

	• Mapa conceitual: analisar os processos de evolução de softwares existentes no mercado nas diversas vertentes. • Aplicação de questionário curto sobre o assunto da aula. • Atividade de fixação individual extraclasse		9a edição. Capítulo 9. Pearson Addison Wesley. 2011.

16. MANUTENÇÃO DE SOFTWARE
- Manutenção de programas.
- Gerenciamento de sistemas legados.

Objetivos de Aprendizagem	Competências Relacionadas
1. Diferenciar os tipos de manutenção de software. 2. Avaliar os sistemas legados de forma a serem descartados, mantidos, passados por reengenharia ou substituídos.	I II XII XIV

Estratégias de Ensino	Avaliação Formativa	Recursos
QUIZ / aula expositiva dialogada / trabalho Sequência sugerida: • Apresentação dos objetivos de aprendizagem • Levantamento dos conhecimentos prévios: QUIZ • Aula expositiva dialogada: apresentação sobre manutenção de software e gerenciamento de sistemas legados. • Manutenções: Corretiva, Preventiva, Adaptativa e Perfectiva • Leis de Lehman • Análise dos Custos de Manutenção de Software • Atividade prática: analisar como softwares comerciais grandes são mantidos e quando se tornam legados. • Aplicação de questionário curto sobre o assunto da aula. • Atividade de fixação individual extraclasse	Kahoot ou Socrative.	PRESSMAN, R. , MAXIM, B. Engenharia de Software, Capítulo 36 e 37, 8th edição. AMGH, 01/2016 SOMMERVILLE, I. Engenharia de Software. 9a edição. Capítulo 9. Pearson Addison Wesley. 2011.

17

18

19

20

21

22

Essas unidades estão disponíveis para acomodar, quando presente, e não necessariamente nesta ordem:
- Aplicação de avaliações
- Revisão ou reforço de conteúdos mediante avaliação da performance da turma
- Feriados e eventos fortuitos

AVALIAÇÃO

A Média Final (MF) da disciplina considera os seguintes elementos e valores:

N1	N2
A1 – Avaliação(ões) a ser(em) definida(s) de acordo com os objetivos de aprendizagem **[nota de 0 a 10]**	**A2 – Avaliação(ões)** a ser(em) definida(s) de acordo com os objetivos de aprendizagem (9,0 pontos) + **APS –** Atividade Prática Supervisionada (1,0 ponto) **[nota de 0 a 10]** **OU** **SUB –** Avaliação Substitutiva **[nota de 0 a 10]**

A Média Final (MF) é calculada por meio da média ponderada das duas notas, N1 e N2, com peso, respectivamente de 40% e 60%, resultante da seguinte equação:

MF = (N1*0,4) + (N2*0,6)

Para aprovação, a Média Final deverá ser igual ou superior a 6,0 (seis), além da necessária frequência mínima de 75% nas aulas.

O estudante que não realizar a A2 ou não atingir a média final 6,0 (seis) na disciplina, poderá realizar uma Avaliação Substitutiva (SUB), cuja nota substituirá a nota de A2 obtida, caso seja maior.

BIBLIOGRAFIA BÁSICA

PRESSMAN, R. , MAXIM, B. Engenharia de Software, 8th edição. 2016

PFLEEGER, S. L. **Engenharia de Software - Teoria e Prática**. 2a edição. Pearson Addison Wesley. 2004.

SOMMERVILLE, I. **Engenharia de Software**. 9a edição. Pearson Addison Wesley. 2011.

BIBLIOGRAFIA COMPLEMENTAR

LARMAN, Craig. **Utilizando UML e Padrões. Bookman**. 2011

FOWLER, Martin. **UML Essencial.** 2011

FILHO, PADUA, Wilson Paula. **Engenharia de Software - Fundamentos, Métodos e Padrões.** 3ª edição. LTC, 2008.

SCHACH, R., S. **Engenharia de Software: Os Paradigmas Clássico & Orientado a Objetos.** 7.ed. São Paulo: AMGH, 2010. 9788563308443. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788563308443/. Accesso em: 03 Apr 2020

LARMAN, Craig. **Utilizando UML e Padrões.** Bookman. 2011

ERL, T. **SOA - Princípio de Design de Serviços**. Pearson Education do Brasil. 2009.

MEDEIROS, E. **Desenvolvendo Software com UML 2.0 Definitivo**. Pearson Makron Books. 2004.

LEE, R. C. TEPFENHART, W. M. **UML e C++ - Guia Prática de Desenvolvimento Orientado a Objeto**. Makron Books. 2001.

LEE, V. SCHNEIDER, H. SCHELL, R. **Aplicações Móveis - Arquitetura, Projeto e Desenvolvimento**. Pearson Makron Books. 2005.

PAGE-JONES, M. **Fundamentos do Desenho Orientado a Objeto com UML**. Pearson Makron Books. 2000.

GUIA DO SCRUM BR - https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf

CONHECIMENTO EM SCRUM (SBOK) - https://www.scrumstudy.com/SBOK/SCRUMstudy-SBOK-Guide-2016-Portuguese.pdf