DIRETORIA DE ENSINO
DEPARTAMENTO DE TELEMÁTICA
COORDENAÇÃO DO CURSO TÉCNICO INTEGRADO EM INFORMÁTICA

PROGRAMA DE UNIDADE DIDÁTICA – PUD

DISCIPLINA: MÉTODOS E FERRAMENTAS PARA DESENVOLVIMENTO DE SOFTWARE		
Código: 01.106.45		
Carga Horária Total:	Teórica: 40	Prática: 40
CH – Prática como Componente Curricular do ensino:		
Número de Créditos:	4,0	
Pré-requisitos:	01.106.25	
Semestre:	4	
Nível:	Técnico	

EMENTA

Noções fundamentais do ciclo de desenvolvimento do *software*.
Métodos, ferramentas e *frameworks* usados no ambiente de produção de *software*.

OBJETIVOS

Familiarizar-se com os termos empregados em ambientes de desenvolvimento de *software*. Entender os princípios dos Métodos ágeis e da Modelagem de domínio e modelagem conceitual com UML. Compreender todos os estágios pelos quais o software deve passar, antes de entrar em produção, preparando o ambiente onde o desenvolvimento ocorrerá. Configurar e executar um *pipeline* de integração e entrega contínua, iniciando com ferramentas de controle de versão, passando por ferramentas de testes e terminando com o *deploy*. Aplicar padrões de projetos em sua prática de desenvolvimento.

PROGRAMA

1. Introdução ao Ciclo de desenvolvimento de software (2h)
2. Métodos ágeis (2h)
    - SCRUM
3. Modelagem de domínio e modelagem conceitual usando UML (4h)
    - a linguagem UML
    - diagrama de classe
    - conversão de diagrama para código
4. *Containers* e ambiente de desenvolvimento (4h)
    - arquivos de configuração
    - geração de imagens
    - executando o container
5. Testes unitários e de integração (8h)
6. Sistemas de controle de versão (8h))
7. Ferramentas de Integração contínua e entrega contínua (8h)
8. Padrões de projeto (MVC, Singleton, Observer, Facade, etc) (4h)

METODOLOGIA DE ENSINO

Aulas expositivas (50%) e aulas práticas em laboratórios (50%)

RECURSOS

- Quadro-branco
- Projetor
- Computadores com acesso à Internet e ambiente de virtualização instalado (Docker, Vagrant, etc.)

AVALIAÇÃO

Provas para avaliar os conhecimentos teóricos e a avaliação do desempenho do aluno nas práticas.

BIBLIOGRAFIA BÁSICA

1) Fundamentos do Desenho Orientado a Objeto com UML (Page-Jones, Meilir)

2) Rossel, Sander. Continuous Integration, Delivery, and Deployment: Reliable and faster software releases with automating builds, tests, and deployment. Packt Publishing Ltd, 2017.

3) GAMMA, Erich, HELM, Richard; JOHNSON, Ralph, VLISSIDES, John."Padrões de Projeto: soluções reutilizáveis de software orientado a objetos". 1. ed. Porto Alegre: Bookman, 2000.

BIBLIOGRAFIA COMPLEMENTAR

1) Fowler, Martin, and Matthew Foemmel. "Continuous integration." Thought-Works) http://www. thoughtworks. com/Continuous Integration. Pdf 122 (2006): 14.

2) The Docker Book: Containerization Is the New Virtualization (James Turnbull)

3) STELLMAN E GREENE, Andrew e Jeniffer, Learning Agile – Understanding Scrum, XP, Lean and kanban. 1º Edição. Sebastopol, O 'Reilly, 2015.

4) Lewis, William E. Software testing and continuous quality improvement. Auerbach publications, 2017.

5) Scott, Kendall, and Martin Fowler. UML Distilled Second Edition A Brief Guide to the Standard Object Modeling Language. Addison Wesley, 2017.

Coordenador do Curso	**Setor Pedagógico**