PLANO DE ENSINO

Data de Emissão: 21/07/2016

Instituto de Informática

Departamento de Informática Aplicada

Dados de identificação

Disciplina: ENGENHARIA DE SOFTWARE N

Período Letivo: 2016/2 **Período de Início de Validade:** 2016/2

Professor Responsável pelo Plano de Ensino: KARIN BECKER

Sigla: INF01127 **Créditos:** 4 **Carga Horária:** 60

Súmula

A engenharia de software. A crise do software. A produção de software. O ciclo de vida do software. A especificação de requisitos. O projeto de software. A implementação, o teste e a documentação do software.

Currículos

Currículos	Etapa Aconselhada	Natureza
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO	5	Obrigatória
ENGENHARIA DE COMPUTAÇÃO	5	Obrigatória

Objetivos

O desenvolvimento da disciplina busca dar ao aluno, ao final do semestre, condições de:

1. Compreender a importância, contexto e conceitos da Engenharia de Software;
2. Compreender as fases, atividades e papéis envolvidos no ciclo de vida de um software;
3. Aplicar técnicas de modelagem para as principais atividades do ciclo de vida de um software;
4. Dentro de uma metodologia de desenvolvimento de software, relacionar as principais técnicas de modelagem às atividades do ciclo de vida de um software;
5. Conhecer tendências em Engenharia de Software.

Conteúdo Programático

Semana: 1 a 2

Título: Introdução a Engenharia de Software: Conceitos, ciclo de vida e processos

Conteúdo:

1. Introdução à Engenharia de Software
1.1. Programas vs. Engenharia de Sistemas
1.2. Crise de Software
1.3. Conceitos e Princípios básicos da Engenharia de Software
2. Processo de Software
2.1. Atividades do Processo
2.2. Modelos de Processo de Software
2.3. Rational Unified Process

Semana: 3

Título: Modelos de Software e UML

Conteúdo:

3. Modelagem no Desenvolvimento de Software
3.1. Abstração e Modelagem de Software
3.2. Visão Geral da Unified Modeling Language (UML)
3.2.1. Visão Geral dos Diagramas
3.2.2. Modelos vs. Diagramas

Semana: 4 a 7

Título: Requisitos

Conteúdo:

4. Engenharia de Requisitos
4.1. Requisitos: conceito
4.2. Processo: Atividades, Papéis e Artefatos
4.3. Modelo de Casos de Uso
4.3.1. Diagrama de Casos de Uso

	4.3.2. Descrição de Casos de Uso
	4.3.3. Diagrama de Atividades
Semana:	8
Título:	Análise
Conteúdo:	5. Análise
	5.1. Análise: conceitos
	5.2. Processo: Atividades, Papéis e Artefatos
	5.3. Modelos de Análise
	5.3.1. Diagrama de Estados
	5.3.2. Diagrama de Classes (nível de análise)
Semana:	9 a 12
Título:	Projeto e Implementação
Conteúdo:	6. Projeto e Implementação
	6.1. Projeto: Conceitos
	6.2. Processo: Atividades, Papéis e Artefatos
	6.3. Projeto Arquitetural
	6.3.1. Diagrama de Pacotes e de Componentes
	6.3.2. Padrões Arquiteturais
	6.4. Projeto Detalhado
	6.4.1. Princípios
	6.4.2. Diagrama de Classes (nível de projeto)
	6.4.3. Diagramas de Interação (Sequência e Colaboração)
	6.5. Implementação
Semana:	13
Título:	Verificação e Validação
Conteúdo:	7. Verificação e Validação
	7.1. Verificação e Validação: Conceitos
	7.2. Verificação em todas atividades de desenvolvimento
	7.3. Artefatos
	7.3.1. Plano de teste
	7.3.2. Casos de Teste
	7.4. Tipos de Teste
	7.5. Ambientes de Apoio a Teste
Semana:	14
Título:	Evolução de Software
Conteúdo:	8. Evolução de Software
	8.1. Processos de Evolução
	8.2. Dinâmica da Evolução
	8.3. Manutenção de Software
	8.4. Gerenciamento de Sistemas Legados
Semana:	15
Título:	Desenvolvimento Ágil de Software
Conteúdo:	9. Desenvolvimento Ágil de Software
	9.1. Conceitos e Princípios
	9.2. Métodos ágeis

Metodologia

A disciplina é apresentada em aulas teórico-práticas, nas quais são combinadas a apresentação de conceitos e técnicas, com a aplicação destes pelos alunos sobre os exemplos, exercícios e os trabalhos extra-classe. As 60 horas previstas para atividades teóricas e práticas indicadas neste Plano de Ensino incluem 30 encontros de 100 minutos de duração (2 períodos de 50 minutos por

encontro, 2 encontros por semana, durante 15 semanas), num total de 3.000 minutos, e mais 10 horas (600 minutos) de atividades autônomas, realizadas sem contato direto com o professor, correspondentes a exercícios e trabalhos extraclasse O moodle será utilizado como arcabouço de ensino, concentrando os planos de aula, o material didático disponibilizado pelo professor, comunicação e tarefas.

Carga Horária

Teórica: 45

Prática: 15

Experiências de Aprendizagem

Os alunos desenvolverão exercícios de fixação para os exercícios de modelagem, bem como dinâmicas que permitam compreender conceitos da engenharia de software e do desenvolvimento de software em equipes.

Os alunos deverão também desenvolver um trabalho em grupo, que consta da especificação de um sistema, sobre os quais serão praticadas técnicas e experiências típicas de um ambiente de desenvolvimento e manutenção de software. Com isso, experimentarão, na prática, as dificuldades inerentes ao processo de modelagem de requisito nas diferentes etapas de desenvolvimento de software. Nas atividades de modelagem utilizarão os modelos da UML (Unified Modelling Language) para produção de artefatos. Deverão experimentar atividades de desenvolvimento de software em times, e uso de ferramentas CASE.

Critérios de avaliação

• Provas: Serão realizadas 2 (duas) provas, que ocorrem conforme calendário da disciplina em datas pré-definidas. As provas envolvem o conteúdo de toda disciplina ministrado até a prova. A avaliação de cada prova é divulgada em até 3 semanas após realização da prova, ou 72 horas antes da prova de recuperação. Peso 4 sobre o total, a razão do peso 2 para cada prova.

• Trabalhos Práticos: Serão realizados trabalhos práticos, organizados em sua maioria como atividades extraclasse. A realização dos trabalhos práticos é caracterizada com atividade autônoma, mas estão previstas aulas de acompanhamento com o professor para resolver dúvidas da elaboração do trabalho. Os trabalhos práticos serão realizados em grupo e seus desenvolvimentos (parciais e/ou final) enviados via moodle nas datas indicadas. O conjunto dos trabalhos práticos tem peso 5 sobre o total. Haverá apresentações (parciais ou final) do trabalho, e seu resultado final divulgado até 72 horas antes da prova de recuperação.

• Participação: Incluir a avaliação sobre realização/participação de exercícios e dinâmicas propostos, aulas de laboratório, posicionamento quanto a conteúdo e dúvidas, qualidade de participação em aula e motivação durante o desenvolvimento dos trabalhos, e assiduidade do aluno. Peso 1 sobre o total.

• Formação do Conceito Final: O conceito final do aluno será atribuído levando-se em consideração as provas (ou trabalhos teóricos), trabalhos práticos e participação, nos pesos acima definidos.

A média ponderada das provas, trabalhos e avaliação do professor será convertida em conceito, mediante escala abaixo:

Nota >= 9,0 = A

Nota >= 7,5 e < 9,0 = B

Nota >= 6,0 e < 7,5 = C

Nota < 6,0 = D

Observações: Somente será calculada a média geral daqueles alunos que tiverem obtido um índice de frequência às aulas igual ou superior a 75% das aulas previstas. Aos que não satisfizerem este requisito, será atribuído o conceito FF (Falta de Frequência).

Atividades de Recuperação Previstas

O aluno que obtiver conceito final D pode realizar uma prova de recuperação versando sobre todo o conteúdo da disciplina. Se a nota obtida nessa prova for igual ou superior a 6,0, o conceito mudará para C.

Observações: Para poder realizar a prova de recuperação, o aluno deve ter realizado ao menos uma das provas/trabalhos teóricos E ter entregue o trabalho prático. Os que não se enquadrarem nesta situação permanecerão com conceito D.

Bibliografia

Básica Essencial

Larman, C. Utilizando UML e Padrões - Uma Introdução à Análise e ao Projeto Orientados a Objetos. Porto Alegre: Bookman, 2008. ISBN 97885600315281.

Sommerville, Ian. Engenharia de software. São Paulo: Pearson Education, 2011. ISBN 9788579361081.

Básica

Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal. Pattern-oriented software arquitecture : a system of patterns. Willey, 1996. ISBN 10 0471958697.

Pressman, Roger. Engenharia de Software: Uma Abordagem Profissional.. McGraw-Hill, 2011.

Complementar

Kent Beck. Extreme Programming Explained: Embrace Change. Boston: Addison-Wesley, 2005. ISBN 0321278658.

Mauro Pezee, Michal Young. Software Testing and Analysis: Process, Principles and Techniques. Willey, 2007. ISBN 0471455938.

Mike Cohn. Desenvolvimento de Software Com Scrum. Bookman, 2011. ISBN 8577808076.

Ron Patton. Software Testing. Sams Publishing;, 2005. ISBN 0672327988.

Outras Referências

Não existem outras referências para este plano de ensino.

Observações

A disciplina é normalmente oferecida em duas turmas, no 1º semestre de cada ano, às 3as e 5as feiras, das 10:30 às 12:20 e das 13:30 ás 15:20.

No 2º semestre de cada ano é oferecida apenas em uma turma, nas 3as e 5as , das 13:00 as 15:20 .