SERVIÇO PÚBLICO FEDERAL
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

PROJETO PEDAGÓGICO DE CURSO

Engenharia de Software

Diretor Eduardo Simões de Albuquerque

Vice-Diretor Sérgio Teixeira de Carvalho

Coordenador do curso Cássio Leonardo Rodrigues

Vice-coordenador Edmundo Sérgio Spoto

Coordenador de estágio Edmundo Sérgio Spoto

Coordenadora de monitoria Elisângela Silva Dias

Núcleo Docente Estruturante: Adriana Silveira de Souza
Cássio Leonardo Rodrigues
Edmundo Sérgio Spoto
Fábio Nogueira de Lucena (presidente)
Fernando Marques Federson
Juliana Pereira de Souza-Zinader
Juliano Lopes de Oliveira
Marcelo Ricardo Quinta
Plínio de Sá Leitão Júnior
Renata Dutra Braga

Goiânia – Goiás – Brasil
22 de Setembro de 2016

Sumário

Sumário 1
Lista de siglas 6
Agradecimentos 7
Apresentação 8
Identificação 9
Contexto 10
Da aptidão do Instituto de Informática 10
Da carência de mão de obra 11
Dos indicadores socioeconômicos regionais 11
Exposição de motivos 13
Objetivos 14
Objetivo geral 14
Objetivos específicos 14
Expectativa da formação do profissional 15
Perfil do egresso 15
Habilidades do egresso 15
Princípios norteadores 18
Formação Ética e Função Social do Profissional 19
Formação Técnica 19
Estratégia da definição das disciplinas 20
Articulação entre Teoria e Prática Profissional 21
Interdisciplinaridade 21
Disciplina “Prática em Engenharia de Software” 22
Atividades supervisionadas 22
Núcleo Docente Estruturante (NDE) 23
Política e gestão de estágio não obrigatório 24
Trabalho de Conclusão de Curso (TCC) 25
Integração ensino, pesquisa e extensão 26
Avaliação do processo de ensino e aprendizagem 27
Avaliação do projeto de curso 29
Política de qualificação 29
Requisitos legais e normativos 31
Libras 31
Política de educação ambiental 31

Planejamento das disciplinas 32
Programa institucional 32
Resolução Nº 1 de 17 de junho de 2004 33
Disciplinas de graduação obrigatórias 34
Programa institucional 34
Diretrizes nacionais para Educação em Direitos Humanos 35
Proteção dos direitos da pessoa com Transtornos do Espectro Autista 35
Estrutura Curricular 36
Matriz curricular 36
Núcleo livre 38
Atividades complementares 38
Distribuição da carga horária 38
Fluxo sugerido 39
Pré-requisitos 43
Disciplinas de formação básica (obrigatórias) 45
Computação e Sociedade 45
Introdução à Programação 45
Cálculo 1A 46
Fundamentos de Matemática para Computação 46
Arquitetura de Computadores 47
Algoritmos e Estruturas de Dados 1 47
Probabilidade e Estatística A 47
Álgebra Linear 48
Lógica Matemática 48
Programação Orientada a Objetos 49
Algoritmos e Estruturas de Dados 2 49
Linguagens e Paradigmas de Programação 49
Engenharia de Software 50
Análise e Projeto de Algoritmos 50
Interação Humano-Computador 51
Banco de Dados 51

Projeto de Software....52
Disciplinas optativas....53
Sistemas Operacionais....53
Pesquisa Operacional....53
Linguagens Formais e Autômatos....53
Redes de computadores....54
Introdução à Língua Brasileira de Sinais....54
Compiladores....54
Sistemas Distribuídos....55
Disciplinas específicas de Engenharia de Software....56
Condições mínimas....56
Construção de Software....57
Modelagem de Software....59
Processos de Software....59
Engenharia de Sistemas....60
Design de Software....61
Domínios de Software....62
Processos de Qualidade de Software....63
Gerência de Projeto de Software....64
Software Concorrente e Distribuído....65
Experiência do Usuário de Software....66
Arquitetura de Software....67
Requisitos de Software....68
Governança e Gestão de Serviços de Software....69
Software para Sistemas Ubíquos....70
Software para Persistência de Dados....71
Padrões de Arquitetura de Software....72
Teste de Software....73
Metodologia e Experimentação em Engenharia de Software....74
Mercado e Economia de Software....74
Prática em Engenharia de Software....75

Referências 79
Apêndice A 81
Tabelas de equivalência 82
Apêndice B 90
Disciplinas de formação básica (obrigatórias) 90
Computação e Sociedade 90
Introdução à Programação 91
Cálculo 1A 91
Fundamentos de Matemática para Computação 92
Arquitetura de Computadores 93
Algoritmos e Estruturas de Dados 1 93
Probabilidade e Estatística A 94
Álgebra Linear 95
Lógica Matemática 95
Programação Orientada a Objetos 96
Algoritmos e Estruturas de Dados 2 96
Linguagens e Paradigmas de Programação 97
Engenharia de Software 98
Análise e Projeto de Algoritmos 98
Interação Humano-Computador 99
Banco de Dados 100
Projeto de Software 100
Disciplinas optativas 101
Pesquisa Operacional 101
Linguagens Formais e Autômatos 102
Sistemas Operacionais 102
Compiladores 103
Redes de computadores 103
Introdução à Língua Brasileira de Sinais 104
Sistemas Distribuídos 105
Metodologia e Experimentação em Engenharia de Software 105
Mercado e Economia de Software 106

Disciplinas específicas de Engenharia de Software 107
Engenharia de Sistemas 107
Construção de Software 108
Modelagem de Software 108
Processos de Software 109
Domínios de Software 110
Processos de Qualidade de Software 110
Gerência de Projeto de Software 111
Design de Software 112
Governança e Gestão de Serviços de Software 113
Arquitetura de Software 113
Requisitos de Software 114
Experiência do Usuário de Software 115
Software Concorrente e Distribuído 115
Padrões de Arquitetura de Software 116
Teste de Software 117
Software para Sistemas Ubíquos 117
Software para Persistência de Dados 118
Prática em Engenharia de Software 118

Lista de siglas

ACM	Association for Computing Machinery
BES	Bacharelado em Engenharia de Software
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CC	Ciências da Computação (bacharelado)
COMTEC	Comunidade Tecnológica de Goiás
CNPq	Conselho Nacional de Desenvolvimento Científico e Tecnológico
DCN	Diretrizes Curriculares Nacionais
ENADE	Exame Nacional de Desempenho de Estudantes
FAPEG	Fundação de Apoio à Pesquisa do Estado de Goiás
IEEE	Institute of Electrial and Electronics Engineers
IME	Instituto de Matemática e Estatística
INF	Instituto de Informática
Letras	Faculdade de Letras
MEC	Ministério da Educação
NDE	Núcleo Docente Estruturante
P&D	Pesquisa e Desenvolvimento
PPC	Projeto Pedagógico de Curso
REUNI	Reestruturação e Expansão das Universidades Federais
SI	Sistemas de Informação (bacharelado)
TAE	Técnico-Administrativo em Educação
TIC	Tecnologia da Informação e Comunicação
UFG	Universidade Federal de Goiás

Agradecimentos

A Deus, por tudo.

Apresentação

O curso de Bacharelado em Engenharia de Software (BES) é oferecido pelo Instituto de Informática (INF) da Universidade Federal de Goiás (UFG). A primeira versão do Projeto Pedagógico do Curso (PPC) foi criada em 2008, a segunda versão, registrada nesse documento, é fruto de uma reflexão de dois anos do Núcleo Docente Estruturante (NDE), que reinventou o BES.

A busca pela qualidade é a motivação (p. 13) dessa segunda versão, que só pode ser adequadamente compreendida com o detalhamento da história da criação do curso e do cenário no qual está inserido (p. 10), o que lembra Paulo Freire, segundo o qual "toda leitura de texto pressupõe uma rigorosa leitura do contexto".

Não houve alteração relevante dos objetivos do curso (p. 14), nem tampouco do perfil do egresso (p. 15), contudo, ambos foram refinados. Os requisitos legais e normativos foram ampliados pelas diretrizes curriculares nacionais (que não existiam quando a primeira versão foi produzida). Em particular, a estratégia para atender a educação das relações étnico-raciais e para o ensino de história e cultura afro-brasileira e africana foi revista em sua totalidade.

A estrutura curricular (p. 35) sofreu mudanças significativas. Em particular, o fluxo sugerido, nos quatro primeiros períodos, inclui apenas disciplinas de formação básica em computação. A disciplina "Prática em Engenharia de Software" (p. 75) é o elemento integrador do curso, com carga horária de 320 horas. Em particular, as disciplinas específicas de Engenharia de Software são acompanhadas de uma seção adicional, "condições mínimas" (p. 56), que estabelece competências a serem demonstradas/adquiridas pelo estudante em cada uma delas.

A busca por um curso de Engenharia de Software melhor é uma atribuição contínua do NDE estabelecida nesse PPC por meio de princípios (p. 23). Esse compromisso é oportuno para estabelecer claramente a responsabilidade pela gestão do presente PPC pelo NDE, sem o qual esse texto resume-se a um registro de intenções.

Identificação

Nome do curso	Engenharia de Software
Grau acadêmico	Bacharelado
Título do egresso	Bacharel(a) em Engenharia de Software
Área de conhecimento (CNPq)	Ciências Exatas e da Terra (1.00.00.00-3) Ciência da Computação (1.03.00.00-7) Metodologia e Técnicas da Computação (1.03.03.00-6) **Engenharia de Software (1.03.03.02-2)**
Modalidade	Presencial
Local de oferta	Instituto de Informática (UFG) Alameda Palmeiras, Quadra D, Câmpus Samambaia Goiânia (GO), CEP 74690-900
Reconhecimento	Portaria N° 819 de 29 de outubro de 2015. Número de registro no e-MEC: 201307356.
Responsável	Instituto de Informática (INF)
Unidades executoras	Instituto de Informática (INF) Instituto de Matemática e Estatística (IME) Faculdade de Letras (Letras)
Número de vagas	40 vagas anuais
Carga horária	3200
Duração do curso em semestres	9 semestres
Turno de funcionamento	Noturno
Forma de ingresso	Sistema de Seleção Unificada (SiSU). Em caso de existência de vagas é possível o ingresso através: (a) de transferência de outras instituições de ensino superior; (b) portadores de diploma ou (c) reingresso. Essas opções dependem de processo seletivo específico na UFG.

Contexto

O Bacharelado em Engenharia de Software (BES) existe e se mantém em um contexto no qual se destaca a aptidão do Instituto de Informática, a carência de mão de obra apta a desenvolver softwares de qualidade e a importância local do curso, detalhados nas seções seguintes.

Da aptidão do Instituto de Informática

O Instituto de Informática da UFG possui sólida experiência no ensino superior em Computação, iniciada em 1983, com a primeira turma do curso de Bacharelado em Ciências da Computação (CC). A partir de 2009 foram criados mais dois cursos de graduação: o Bacharelado em Engenharia de Software (BES) e o Bacharelado em Sistemas de Informação (SI). O INF também é responsável por dezenas de disciplinas da área de Computação ministradas para diversos cursos de graduação de várias unidades de ensino da UFG.

Na pós-graduação *lato sensu* o INF já ofertou dezenas de edições dos mais variados cursos de especialização na área de Computação. Na pós-graduação *stricto sensu* o INF oferece, desde 2001, o Mestrado em Ciência da Computação. A partir de 2010 o INF passou a oferecer o Doutorado em Ciência da Computação.

Essa história perfaz mais de três décadas de ensino, com milhares de egressos, alguns deles ocupando cargos de destaque em universidades, empresas e organizações tanto no Brasil quanto no exterior.

O INF construiu, ao longo desse período, um consistente envolvimento com empresas e organizações de Tecnologia da Informação e Comunicação (TIC). O Apoema [APOEMA] é o órgão do INF responsável pela cooperação e interação com organizações externas, com foco em soluções inovadoras.

Um número significativo de projetos de Pesquisa e Desenvolvimento (P&D) já foi realizado pelo INF em parceria com organizações locais, e alguns com empresas multinacionais (como Dell e HP, por exemplo). O INF também já participou de vários projetos de inovação tecnológica financiados pela Fundação de Apoio à Pesquisa do Estado de Goiás (FAPEG).

Da carência de mão de obra

Um dos motivos para a existência do BES é a necessidade de formar recursos humanos em Engenharia de Software. A demanda por profissionais especializados na produção de software não existe só no Estado de Goiás, nem tampouco apenas no Brasil, mas em todo o mundo.

Apesar da demanda, na ocasião da criação do BES, não era conhecido outro curso com a mesma denominação ou propósito no Brasil. A tradição do INF na socialização da Computação, juntamente com a atuação em Engenharia de Software do seu corpo docente, resultou em proposta pioneira no País de um curso especificamente voltado para a formação de Engenheiros de Software, com ênfase na produção de software. Hoje, o Guia dos Estudantes da Editora Abril lista 28 cursos similares.

A região da Grande Goiânia reúne centenas de empresas de TIC, mas elas participam de forma tímida no mercado de software nacional. Com a disponibilidade de recursos humanos qualificados, essa participação pode crescer, o que é compatível com a dimensão do Estado de Goiás, a oitava economia nacional. Portanto, promover a indústria de produção de software em Goiás significa fomentar o aumento da participação das empresas do estado no mercado de software.

Dos indicadores socioeconômicos regionais

O BES cria a possibilidade de acesso ao ensino superior para aqueles que não podem usufruir do ensino privado. Segundo o IBGE [IBGE], em Goiás, 62.133 estudantes de graduação frequentavam cursos públicos em 2010, enquanto 156.415 estavam matriculados em cursos privados. Ou seja, estudantes em cursos públicos representam menos de 30% do total de estudantes matriculados no ensino superior.

O rendimento nominal médio mensal domiciliar *per capita* de todos os municípios brasileiros revela que, das cidades goianas, a capital do estado é a melhor posicionada, em vigésimo quarto lugar. O município goiano seguinte nesta classificação é Alto Paraíso de Goiás, na posição 158. O terceiro é Jataí, na posição 188. Até esta terceira aparição de municípios goianos, observa-se que

o Estado do Rio Grande do Sul contribui com 57 municípios, São Paulo contribui com 55 municípios e Santa Catarina com 25 municípios.

Quando se analisa o produto interno bruto (PIB) dos municípios brasileiros, dentre os cem maiores, Goiás contribui com apenas 2 municípios (Goiânia e Anápolis). Reunidos, estes dois municípios possuem PIB inferior ao do décimo terceiro colocado, São Bernardo do Campo (SP).

Os valores expostos acima sugerem que a capacidade de financiamento privado do ensino superior em Goiás é inferior à de outros estados. Apesar dessa limitação, menos de 30% do total de estudantes estão matriculados em cursos superiores públicos.

Aqueles que conseguem acesso ao ensino superior, público ou privado, são minoria em Goiás. Segundo o [IBGE], em 2010 Goiás tinha 1.213.946 pessoas com 10 ou mais anos de idade com curso médio completo, enquanto apenas 218.548 estavam matriculadas em curso superior. Ou seja, menos de 20% continuam seus estudos até o ensino superior. Neste censo, Goiás tinha 394.491 cidadãos com curso superior completo em uma população total de 6.003.788, ou seja, 6,5% da população com curso superior. O Estado de São Paulo, por exemplo, apresenta uma taxa superior a 10%. Taxas ainda bem superiores são encontradas em outros países [OECD, 2012].

Além de contribuir para melhoria dos indicadores de educação em Goiás, o BES potencializa o crescimento da economia goiana por meio de alternativa à atual dependência do agronegócio. Essa alternativa fomenta a participação de Goiás em um mercado valioso, além de promover a criação de empregos.

A posição estratégia do BES pode ser esclarecida por meio de iniciativas internacionais. Por exemplo, *TechHire Initiative* e *Computer Scicence for All* são programas americanos lançados nos dois últimos anos, orçados em cerca de 2 bilhões de dólares, com o propósito de promover a capacitação em produção de software nos Estados Unidos.

Exposição de motivos

A dependência da sociedade em relação a softwares continua se expandindo, inclusive em tarefas que antes eram exclusivas dos seres humanos, por exemplo, estacionar um veículo. Isso tem ampliado a demanda por softwares e consequentemente por profissionais que visam atender essa demanda. O conhecimento necessário para produzir software é denominado de Engenharia de Software.

Tradicionalmente esse conhecimento e as habilidades necessárias são adquiridas por egressos de outros cursos de graduação ou até por meio de cursos rápidos. Nesses casos, a apropriação de habilidades ocorre de forma parcial, pois não há espaço em tais cursos para cobrir adequadamente essa extensa área.

A vocação de docentes do INF e o Programa de Reestruturação e Expansão das Universidades Federais (REUNI) resultaram na criação do primeiro curso de graduação (bacharelado) em Engenharia de Software do Brasil, o presente curso, pelo INF/UFG em 27/06/2008, conforme a resolução CONSUNI 10/2008, com o foco na formação de egressos aptos a usar de forma efetiva o conhecimento da Engenharia de Software. Essa primeira versão foi amplamente divulgada [BES1] [BES2].

Dezenas de docentes acumularam lições aprendidas desde o início da primeira turma do curso em março de 2009. A partir de outubro de 2014, o Núcleo Docente Estruturante (NDE) do curso iniciou esforço para refletir essas lições em uma nova versão do PPC. Ao longo de quase meia centena de encontros, o NDE fez uma exaustiva revisão do curso. As lições aprendidas e outras demandas, por exemplo, as diretrizes curriculares e o alinhamento com uma formação básica de qualidade em computação, definida pelo próprio INF, foram acomodadas na presente versão.

Convém esclarecer que na época da criação do BES, sem similar no país, não existiam diretrizes curriculares específicas. Essas diretrizes vieram apenas em 2012 [CNE 2012].

A presente versão não é resultado de alterações superficiais aplicadas à versão anterior, mas de ampla reflexão. Insumos empregados na elaboração da versão anterior foram revisitados. As Diretrizes Curriculares Nacionais (DCN), a versão mais recente do corpo de conhecimento da área e o modelo de competência para o engenheiro de software balizaram os esforços realizados. Projetos pedagógicos de cursos semelhantes, inclusive em outros países, também foram investigados. Esse substancial conjunto de informação foi analisado à luz da experiência dos docentes do próprio curso e do contexto local.

Por fim, o BES obteve nota máxima na avaliação do MEC em 2015, o que estimulou o cuidado com o curso, reforçado pelo Guia dos Estudantes da Editora Abril 2015, no qual, dentre os 28 cursos superiores em Engenharia de Software investigados, dois recebem 5 estrelas, um desses é o curso do presente projeto pedagógico. Em 2016, esse cenário persiste, conforme o Guia dos Estudantes 2016.

Objetivos

Objetivo geral

O BES tem como objetivo

> *Formar profissionais aptos a contribuir efetivamente com a produção de softwares de qualidade seguindo princípios éticos e postura profissional.*

Objetivos específicos

- Reconhecer e valorizar o respeito à diversidade. Conforme se lê na Constituição Federal em seu Art. 3.º, inciso IV: "promover o bem de todos, sem preconceitos de origem, raça, sexo, cor, idade e quaisquer outras formas de discriminação".

- Promover relações étnico-raciais respeitosas, assim como a socialização da história da formação dos brasileiros.

- Ampliar a compreensão acerca de questões ambientais e, ao mesmo tempo, valorizar projetos sustentáveis.

- Contribuir com a demanda da sociedade por softwares de qualidade.
- Conceber e desenvolver produtos inovadores.
- Promover a engenharia de software na indústria de software regional.
- Viabilizar atividade econômica de alto valor (produção de software).
- Criar alternativa econômica para o Estado de Goiás.
- Fortalecer a indústria de software goiana.

Expectativa da formação do profissional

Perfil do egresso

O egresso do BES é um profissional com capacitação sólida em engenharia de software. Essa capacitação advém de formação básica consistente em computação e de abrangente e aprofundada em engenharia de software.

O mercado de atuação do egresso é abrangente, e decorre da dependência da sociedade por software. Adicionalmente, tal dependência tem se expandido com novos produtos, serviços e processos.

A bacharela ou o bacharel em Engenharia de Software é capaz de efetivamente contribuir com equipes na produção de modelos abstratos de software e realizá-los por meio de código de qualidade. Essa contribuição é pautada por postura profissional e conduta ética.

As habilidades do egresso incluem o que é necessário para lidar com requisitos de software, propor uma solução (*software design*), construí-la, testá-la e mantê-la, usando processos adequados.

Habilidades do egresso

Da perspectiva de relações pessoais, o egresso deve ser capaz de:

- Participar de forma harmoniosa, profissional e ética durante a elaboração de produtos de software atribuídos a equipes.
- Liderar ação contínua de formação de sua própria reputação na área.

- Participar da comunicação de ideias com clareza, seja na forma verbal ou escrita.

Da perspectiva técnica, o egresso deve ser capaz de:

- Participar de atividades para eliciar, analisar, especificar, validar e gerenciar requisitos de software.
- Participar da definição da solução de software que atende requisitos, ou seja, da definição da arquitetura e do projeto detalhado desse software.
- Participar da elaboração de modelos de análise e de projeto (*design*) de software.
- Participar da aplicação de técnicas de projeto ao desenvolver softwares concorrentes, distribuídos, orientados a objetos, orientados a eventos e que implementam a persistência de dados.
- Participar da avaliação de *design* de software considerando, dentre outros quesitos: abstração, coesão, acoplamento e encapsulamento.
- Participar da aplicação de técnicas de tratamento de exceção e tolerância a falhas.
- Auxiliar a escolha de processo e de metodologia de *design* de software.
- Auxiliar a revisão de *design* de software
- Participar do emprego de estilos, visões, modelos e padrões de arquitetura de software.
- Participar do projeto de componentes e módulos de software usando modelos, padrões de projeto e notações.
- Auxiliar a execução de análise estática.
- Auxiliar a escolha de processos e modelos apropriados para a construção de software.
- Auxiliar a escolha de linguagens e ferramentas apropriadas para a construção de software.

- Auxiliar a escolha de *frameworks*, plataformas e ambientes de construção de construção de software.
- Participar da construção de software seguindo padrões de gerência de configuração e controle de versão.
- Participar da coleta e do monitoramento de medidas de qualidade de código.
- Participar da criação de *design* detalhado que minimiza a complexidade e melhora a qualidade.
- Participar da criação de código que implementa projetos detalhados.
- Indicar o uso de padrões de projeto.
- Participar da refatoração (*refactoring*) de código.
- Participar da construção de software que segue padrões de código.
- Participar da construção de software que faz uso de técnicas de codificação defensiva.
- Participar da geração de código a partir de modelos de projeto.
- Participar do uso apropriado de ferramentas e técnicas de depuração.
- Participar da criação de testes de unidade.
- Participar da construção de software que satisfaz objetivos de cobertura de testes.
- Participar da construção de software que segue estratégias e processos de integração e implantação.
- Participar da construção e realização de testes de integração.
- Participar da construção de software em equipe ao colaborar com outros membros da equipe.
- Participar de revisões e inspeções.

- Participar da elaboração e implementação de planos de testes.
- Participar da definição de um ambiente de teste e da correspondente implementação.
- Participar da identificação, *design* e implementação de testes.
- Participar da elaboração de relatórios de testes.
- Auxiliar a elaboração e execução de plano de transição de software.
- Auxiliar a elaboração e execução de planos de manutenção de software.
- Auxiliar a execução de atividades de processos de software.
- Auxiliar a definição e personalização de processos de software.
- Auxiliar a implementação e execução de processos de software.
- Auxiliar a coleta de dados para avaliação de processos de software.
- Auxiliar a definição e o desenvolvimento de software que satisfaz objetivos e atributos de qualidade.
- Auxiliar a identificação de medidas de qualidade apropriadas.
- Auxiliar a elaboração de planos e revisões de qualidade.
- Participar da construção de software que segue princípios recomendados para segurança de código.
- Participar da construção de software que segue um plano de gerência de configuração de software.
- Participar da documentação de software.

Princípios norteadores

O BES baseia-se em princípios que fornecem a sustentação necessária para atingir os objetivos do curso (p. 14) e o perfil do egresso (p. 15). Esses princípios são fornecidos abaixo.

Formação Ética e Função Social do Profissional

A vivência de princípios éticos e o conhecimento sobre a responsabilidade social do Engenheiro de Software são elementos imprescindíveis para a formação da postura profissional do egresso do BES.

Ética é assunto de uma das disciplinas do primeiro semestre do curso, "Computação e Sociedade" (p. 45), e volta a ser explicitamente tratada, junto com aspectos profissionais, na última disciplina do curso "Prática em Engenharia de Software" (p. 75). Nesse caso, a postura ética e profissional é condição a ser observada para aprovação nessa disciplina.

O comportamento ético e profissional será trabalhado nas atividades do curso, e não apenas na primeira e última disciplinas. A exigência de uma conduta apropriada em sala de aula, tanto dos docentes quanto dos estudantes, contribui com essa formação. Isso significa promover a qualidade de vida, o respeito à diversidade, o respeito ao meio ambiente. Nesse sentido, não apenas as disciplinas, mas toda e qualquer ação, deve ser pautada pela reflexão do impacto no contexto no qual se insere.

Ainda convém destacar que a área possui um código de ética próprio, Código de Ética e Prática Profissional da Engenharia de Software [ACM/IEEE]. Esse código é uma das bases de fundamentação do curso e deve ser observado no convívio diário do curso.

Formação Técnica

A formação técnica proposta para o BES está fundamentada em bases sólidas: as diretrizes curriculares nacionais [MEC 2012], o guia do corpo de conhecimento em Engenharia de Software [SWEBOK 2014], o corpo de conhecimento recomendado para ser trabalhado em um curso de graduação [SE 2014], e o modelo de competência em Engenharia de Software [SWECOM 2014]. Em consequência, o conteúdo abordado no curso não diverge das orientações nacionais nem internacionais. A organização dele, contudo, é uma "contribuição" para a área, dado que se baseia em experiências aprendidas nos anos anteriores, juntamente com uma visão holística do aprendizado em Engenharia de Software, o que contrasta com a frequente apresentação cartesiana dessa área.

As atividades do curso se aproximam do emprego da Engenharia de Software em projetos reais, no qual os conhecimentos e habilidades não são exigidos de forma fragmentada e isolada. O isolamento de conteúdo é adequado para a classificação do conhecimento que, sem o devido cuidado, pode induzir a definição de disciplinas. A estratégia de definição das disciplinas do BES é fornecida abaixo.

Estratégia da definição das disciplinas

As disciplinas do BES foram definidas com substancial cruzamento de fronteiras de subáreas do conhecimento da Engenharia de Software. A estratégia na qual se define uma disciplina por subárea foi preterida. Convém ressaltar que o corpo de conhecimento da área [SWEBOK 2014] foi extensivamente empregado, mas não como exemplo de organização didática.

Em vez da separação de tópicos induzida pela classificação do conhecimento, o conteúdo (ementa) atribuído a cada disciplina do BES inclui tópicos entre os quais há sinergia. Essa orientação é compatível com a prática da Engenharia de Software e não apenas reconhece, mas também respeita os vínculos entre as suas subáreas. Isso resultou em disciplinas coerentes com a prática da Engenharia de Software.

A Figura abaixo ilustra uma disciplina X qualquer do BES, composta por conteúdo das subáreas A, B e C. Ou seja, o conteúdo de X não está contido estritamente na subárea A, nem tampouco na B ou na C. Em vez disso, reúne e explora a interdependência de conceitos dessas três subáreas.

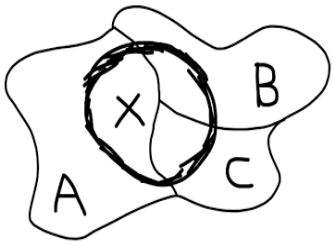


Figura 1 Disciplina do curso baseada em conhecimento de várias subáreas.

As subáreas “requisitos” e “projeto de software”, por exemplo, são contempladas em várias disciplinas do curso e não apenas nas disciplinas nas

quais são o foco principal de interesse. A disciplina que enfatiza testes, por exemplo, inclui aspectos de *design* de software e também de requisitos, assim como aqueles de construção de software. De forma resumida, as disciplinas não são uma projeção cartesiana das 15 subáreas de conhecimento da Engenharia de Software [SWECOM 2014]. Em vez disso, são 18 disciplinas cobrindo reiteradamente várias dessas 15 áreas, sendo que uma delas cobre todas as áreas necessárias para a execução de um projeto real de produção de software (p. 75).

Articulação entre Teoria e Prática Profissional

O perfil do egresso, definido em outra seção desse documento, exige do egresso o envolvimento com o fazer, com o exercício do conhecimento de Engenharia de Software.

Essa articulação é explicitamente estabelecida por meio da seção "Condições mínimas" definida para cada disciplina de Engenharia de Software. Tais condições, em geral, definem o que o egresso pode fazer, em outras palavras, é capaz de realizar com o conhecimento. Ou seja, o que convencionalmente se limita ao conteúdo, ou teoria, deverá ser exercitado, ou prática, e com um nível bem definido de proficiência.

A disciplina "Prática em Engenharia de Software" (p. 75) é um ponto explícito do curso no qual a prática é o elemento principal, perfazendo 320 horas. O fazer, contudo, não está restrito a essa disciplina. A capacidade de realização de atividades de desenvolvimento de software é exigência em boa parte das disciplinas.

Interdisciplinaridade

Produzir software significa, necessariamente, o emprego de pelo menos dois domínios. O domínio da solução, no qual a Engenharia de Software é exercitada e o domínio do problema, que fomenta a existência do software. O domínio do problema é "universal", pois inclui saúde, educação, segurança, governo, finanças e entretenimento, dentre muitos outros. Fazer software, portanto, por si só, exige o contato com outras áreas do conhecimento.

O acesso a outras áreas pode vir das disciplinas do Núcleo Livre (NL). O BES exige para a integralização curricular pelo menos 128 horas de disciplinas do NL. Essas disciplinas são escolhidas pelo estudante dentre todas aquelas oferecidas na UFG. Para ilustrar, no primeiro semestre de 2015 houve oferta de vagas para mais de 300 disciplinas de NL. Ou seja, o conjunto de opções de outras áreas do conhecimento é rico, o que contribui com uma formação ampla do estudante. Convém destacar que tais 128 horas perfazem a quantidade mínima exigida, o estudante pode fazer uso de uma carga horária maior. O mesmo é válido para o estágio não obrigatório e para as atividades complementares. O estudante deve cumprir um mínimo de 192 horas em atividades complementares.

A interdisciplinaridade estimulada pelos elementos citados acima é extra curso. Aquela intracurso é tratada tanto pela definição das disciplinas quanto pela disciplina "Prática em Engenharia de Software" (p. 75). Nesses casos a interdisciplinaridade é compulsória, pois faz parte da própria concepção do curso. Adicionalmente, sem restringir a liberdade metodológica do docente, cabe ao NDE orientar a definição de programas de disciplinas que cultivem a interdisciplinaridade.

Disciplina "Prática em Engenharia de Software"

Essa disciplina de 320 horas tem como objetivo explícito a participação do estudante em um ou mais projetos integradores que usufruem de conhecimento e habilidades adquiridos por todo o curso. Os projetos exigem o contato com problemas reais, o que exige contato tanto com o conhecimento quanto profissionais de outras áreas. Consulte detalhes na página 75.

Atividades supervisionadas

De acordo com a Resolução CNE/CES 03/2007 de 2 de julho de 2007, cabe às Instituições de Educação Superior, respeitando o mínimo dos duzentos dias letivos de trabalho acadêmico efetivo, a definição da duração da atividade acadêmica ou do trabalho discente efetivo, o que compreende: (a) preleções e aulas expositivas e (b) atividades práticas supervisionadas, tais como laboratórios, atividades em biblioteca, iniciação científica, trabalhos

individuais e em grupo, práticas de ensino e outras atividades no caso das licenciaturas.

O BES divide cada hora de atividade acadêmica em 45 minutos de preleções e aulas expositivas e 15 minutos de atividades práticas supervisionadas. O planejamento de cada hora deve estar devidamente registrado no plano de cada disciplina. Em particular, o plano deve incluir de forma clara as atividades práticas supervisionadas.

Núcleo Docente Estruturante (NDE)

Em vez de atribuir atividades ao NDE, o NDE adota os seguintes princípios:

- O NDE possui duas prioridades. A maior prioridade é colocar em prática o Projeto Pedagógico do Curso (PPC). A prioridade seguinte é mantê-lo relevante.
- Qualquer questão que diz respeito às prioridades é do interesse do NDE e pode resultar em ação.
- Toda ação necessariamente deve gerar valor e estar alinhada com as prioridades.
- Uma ação constante e preconcebida é zelar pelos princípios aqui descritos.
- O NDE entende que o diálogo é necessário e que a diversidade é natural, assim como o confronto de ideias.

Esses princípios definem que todo o conteúdo do presente PPC e a aplicação dele está no raio de atuação do NDE. O que inclui:

- Acompanhar a execução das ações de ensino, pesquisa e extensão pertinentes ao curso;
- Acompanhar a avaliação das ações do curso;
- Emitir opinião sobre ações pertinentes ao curso;
- Monitorar o desempenho dos estudantes;
- Promover ações que possam reduzir reprovações;
- Monitorar resultados dos trabalhos de conclusão de curso;

- Monitorar ações de estágios dos estudantes do curso;
- Acompanhar e se pronunciar acerca de reclamações pertinentes ao curso;
- Avaliar de forma contínua o PPC do curso;
- Acompanhar e promover a qualificação de docentes;
- Acompanhar e promover a qualificação do corpo de técnico-administrativos.

O NDE, portanto, não se apresenta como órgão ou mecanismo deliberativo, mas consultivo. Reúne docentes que continuamente refletem sobre questões pertinentes ao curso e, em consequência, definem ações que assistem, apoiam e fomentam a qualidade do curso, bem como contribuem com a execução dessas ações.

A atuação do NDE deve ser realizada em estreita interação com a coordenação do curso. Entretanto, não é nem pode ser vista como meio para auxiliar a realização de fluxos administrativos (atribuição da coordenação do curso).

Convém observar que as atribuições estabelecidas acima estão em conformidade com a resolução da Comissão Nacional de Avaliação da Educação Superior (CONAES), 01/2010 de 17 de junho de 2010, a qual "normatiza o núcleo docente estruturante", e esclarecido pelo parecer CONAES 04/2010 de 17 de junho de 2010.

Política e gestão de estágio não obrigatório

O estágio do BES constitui-se em um mecanismo de complementação de conhecimento e aperfeiçoamento de habilidades, além de oportunidade de prática em Engenharia de Software. O convívio com profissionais, obrigações, hierarquias e processos onde o estágio se desenvolve, resulta em oportunidade valiosa para a formação profissional do egresso.

O estágio do BES não é de caráter obrigatório, ficando a critério do estudante realizá-lo ou não, desde que o mesmo esteja regularmente matriculado no curso. Ou seja, caracteriza estágio curricular não obrigatório. Adicionalmente,

esse estágio deve ser realizado a partir do terceiro período do curso, ou após a conclusão de pelo menos 640 horas da carga horária do curso.

Convém observar que o estágio está restrito às empresas devidamente conveniadas com a UFG ou que se utilizam de agentes de integração conveniados. Adicionalmente, dois outros papéis são obrigatórios no estágio: (a) supervisor (no local do estágio) e (b) orientador (professor do curso).

Durante o estágio, que não pode ultrapassar 24 meses em mesmo local, o estudante deverá apresentar o Termo de Compromisso, o Plano de Estágio, além da frequência e dos relatórios semestrais.

Os documentos citados acima, bem como outros detalhes do estágio, em conformidade com a Lei 11.788/2008, são definidos pelo *Regulamento de Estágio de Curso do Bacharelado em Engenharia de Software.*

Trabalho de Conclusão de Curso (TCC)

O Trabalho de Conclusão de Curso (TCC) no BES é um componente curricular obrigatório regulado pelas *Normas e Procedimentos de Trabalho de Conclusão de Curso do Bacharelado em Engenharia de Software.*

O TCC é desenvolvido na última etapa da graduação, sob a orientação de um professor e compreende um relatório e uma apresentação.

O relatório é individual e tem como objetivo a expressão do estudante na forma escrita, a capacidade de analisar, caracterizar, investigar, discutir, implantar, pesquisar, realizar, sintetizar e avaliar, entre outras. Neste sentido, o relatório descreve, de forma crítica, as atividades teórico-práticas e de formação profissional relacionadas ao desenvolvimento do estudante como profissional, em especial, durante a disciplina Prática em Engenharia de Software (p. 75).

A apresentação do relatório, também individual, é realizada como última atividade do curso e tem como objetivo a expressão do estudante, agora na forma oral, das atividades descritas no relatório, em seção pública e para uma banca formada por professores do INF. A banca de professores tem o direito

a um período de arguição e é responsável pela avaliação tanto do relatório quanto da apresentação.

Integração ensino, pesquisa e extensão

A integração do ensino com a pesquisa e a extensão se verifica por meio de atividades complementares, do estágio não obrigatório, de disciplinas do curso e da postura didática dos docentes, conforme comentado abaixo.

As atividades complementares são identificadas em resolução própria, perfazem pelo menos 192 horas e explicitamente incluem ações de extensão e de pesquisa. Por exemplo, divulgação de trabalhos em eventos científicos, participação em projetos de extensão e participação em projetos de pesquisa, dentre outras.

O estágio não obrigatório é um instrumento de integração. As atividades a serem realizadas pelo estudante podem incluir ações de pesquisa.

A disciplina Metodologia e Experimentação em Engenharia de Software oferece a visão exigida para a realização de pesquisas na área. Isso significa não apenas orientar o estudante acerca da elaboração de uma revisão sistemática, mas também da necessidade da expansão do conhecimento da área e dos mecanismos atualmente empregados para tal.

Um contundente exemplo é a disciplina de 320 horas, Prática em Engenharia de Software. Essa disciplina envolve em sua concepção o ensino, a extensão e a pesquisa ao se concentrar em projetos reais executados no ambiente de uma Fábrica de Software. Qualquer que seja o projeto, esse envolverá extensão e/ou pesquisa, naturalmente, ainda em um cenário de aprendizado (ensino).

A integração, contudo, não ocorre apenas por meio dos elementos acima, que seriam pontos de integração localizados e em períodos específicos. A integração deve ocorrer, onde oportuno, em cada ação do curso. Por exemplo, uma postura didática recomendável exige a contextualização de cada aula acerca do que será visto, da repercussão ou relação dela com a indústria de software (sociedade) e do estado da arte corrente. De fato, essa postura não é apenas recomendável, mas exigida por meio das “condições mínimas”

definidas para cada disciplina. Ou seja, a integração do ensino com a pesquisa e a extensão ocorre por toda a extensão do curso.

Avaliação do processo de ensino e aprendizagem

O que é realizado em nome do curso para que o estudante adquira o perfil esperado pode ser avaliado considerando informações geradas externamente e outras internamente. A avaliação do curso pelo MEC e os resultados obtidos pelos estudantes do curso no ENADE são exemplos de insumos para a avaliação do processo de ensino. Internamente, a avaliação do docente pelo discente (instrumento formal e institucionalizado na UFG), o desempenho dos estudantes do curso nas disciplinas e os planos de ensino das disciplinas são fontes valiosas de informação.

O NDE do curso é o principal consumidor dessas informações com o propósito de detectar possíveis melhorias e fomentar a introdução delas. Por exemplo, embora o docente tenha autonomia metodológica ao planejar e executar o plano de ensino de uma disciplina, cabe ao NDE analisar o plano e, quando considerar oportuno, recomendar ajustes. Essas e outras ações devem ser realizadas durante a semana de planejamento pedagógico e administrativo (semana formalmente institucionalizada na UFG), que ocorre no início de cada período letivo. Convém destacar que o plano de ensino de cada disciplina deve ser aprovado formalmente pelo Conselho Diretor da unidade antes do início do período letivo. Adicionalmente, a ação de avaliação não deve estar restrita à semana de planejamento, mas contínua. O objetivo é viabilizar a introdução de uma eventual mudança o mais próximo possível do instante no qual a necessidade foi detectada.

Outra ação relevante do NDE na semana de planejamento pedagógico é apresentar as lições aprendidas no período anterior. Oferecer visibilidade de processos de ensino exitosos, o que é um instrumento de valorização do docente e, ao mesmo tempo, forma de socializar o que produziu bons resultados, também é uma ação a ser executada nessa semana. Enfim, acerca da avaliação do processo de ensino, o NDE possui papel relevante na identificação de dificuldades e busca de alternativas, em comunhão com os docentes do curso.

A avaliação do processo de ensino, da aprendizagem e do curso, dentre outros, são assuntos pertinentes ao NDE (p. 23). De fato, são interconectados. Em vez de estabelecer ações específicas no presente PPC, são fornecidos princípios adotados pelo NDE do curso (p. 23). Ou seja, orientar os estudantes do curso a cada início do período letivo acerca das normas da UFG, de prazos, da meta de integralização curricular de cada um deles, são algumas ações relevantes, tendo em vista tais princípios.

A avaliação da aprendizagem pode ser materializada de várias formas. São comuns avaliações (provas) escritas e individuais, a avaliação por meio de relatórios ou trabalhos escritos e apresentações orais, dentre outras. A avaliação contínua por meio de exercícios realizados ao longo de toda uma disciplina também é uma forma. De fato, vale ressaltar que um método adequado de avaliação em um cenário não o é em outro.

A avaliação do processo de ensino e aprendizagem do BES deve atender, no seu planejamento e na forma contínua de sua execução, o estipulado pelo Regulamento Geral dos Cursos de Graduação (RGCG) da UFG [CEPEC 2012], notadamente em seu Capítulo IV, Seção 1 - “Da verificação da Aprendizagem”.

Cabe ressaltar que o Sistema de Avaliação do BES tem como objetivo primeiro de sua aplicação, por meio de sua Estrutura Curricular (p. 35), utilizando instrumentos pedagógicos norteados pelos Princípios Norteadores (p. 18), permitir que cada estudante adquira o perfil desejado (p. 15).

A avaliação do estudante deve levar em consideração não apenas o atendimento de requisitos técnicos de produtos e processos das diversas disciplinas da Engenharia de Software, mas também demonstrar postura ética e profissional ao desenvolver ações do curso.

Convém destacar que as disciplinas específicas de Engenharia de Software são acompanhadas, cada uma delas, de seção de “Condições mínimas” (p. 56), que identificam elementos obrigatórios a serem satisfeitos pelo estudante para ser considerado aprovado na disciplina. Embora essa seja uma interpretação correta, a intenção é estabelecer um contrato claro do compromisso do estudante e do docente com a disciplina em questão. A avaliação de cada uma dessas disciplinas, portanto, necessariamente deve observar as condições

mínimas estabelecidas. De fato, a condução de toda a disciplina deve ser orientada por tais condições.

Dada a especificidade da disciplina “Prática em Engenharia de Software” (p. 75), o sistema de avaliação também é específico. Nesse caso, as avaliações devem envolver o domínio de processos e a geração de produtos de Engenharia de Software. Em particular, atividades avaliativas podem ser realizadas mesclando teoria e prática em um ambiente real de desenvolvimento de software, no qual a qualidade estabelecida para os entregáveis do projeto em questão é verificada.

Avaliação do projeto de curso

O presente PPC deverá ser revisado formalmente de cinco em cinco anos, em ação proativa, ou em intervalo de tempo menor, na ocorrência de evento que justifique tal decisão.

O NDE é responsável pelas revisões do PPC e, em particular, por contínua vigilância da adequação do PPC ao cenário corrente. Esse esforço se realizada de duas formas. Uma no sentido de fazer com que as orientações do PPC sejam observadas no cotidiano do curso e outra, onde considerado oportuno, na análise de possíveis melhorias para reagir adequadamente às mudanças do ambiente.

Ao acompanhar o curso, o NDE continuamente coleta, detecta e propõe alterações. As demandas podem se originar dos estudantes, dos docentes e de outras fontes como os resultados dos estudantes do curso no Exame Nacional de Desempenho de Estudantes (ENADE).

Política de qualificação

Desde a criação o Instituto de Informática implementa uma política vigorosa de capacitação do seu corpo docente, a qual inclui a meta de que todos os docentes tenham a titulação mínima de doutor. De fato, a liberação de docentes para prosseguirem os seus estudos de pós-graduação é uma prática comum, em consequência, nenhum pedido para afastamento com o propósito de realizar o doutorado foi negado até o momento. Essa política é

acrescentada de regras institucionalizadas para a concessão de afastamento para o pós-doutorado e de licença para capacitação.

A qualificação de docentes do INF também pode ser obtida pela sua participação em congressos, simpósios, dentre outros eventos, quer seja nos papéis de autor de artigo, organizador de evento, membro de comitê de programa ou avaliador de artigo. O Instituto de Informática ainda financia ou co-financia viagens e inscrições de seus docentes em congressos e simpósios importantes, principalmente quando há publicação de artigo.

Em harmonia com a política de capacitação docente, o INF é favorável ao aprimoramento e à capacitação de seu corpo Técnico-Administrativo em Educação (TAE). Normalmente, um TAE faz solicitação ao diretor, encarregado de registrar as intenções de afastamento no plano anual de capacitação e conduzir a apreciação das intenções pelo Conselho Diretor do INF, haja vista que é do interesse da unidade fomentar a especialização e capacitação dos seus profissionais.

É frequente a liberação das atividades de TAE para que possam participar de treinamentos, tanto em cursos esporádicos quanto em programas de pós-graduação. Em tempo, as ações de extensão do INF reservam vagas exclusivas para participação de TAEs, sem necessidade de pagamento (quando é o caso). Do ponto de vista legal, o INF se apoia na Resolução CEPEC 1286/2014 de 2014 [CEPEC 2014], que regulamenta o afastamento de docentes para cursar Mestrado, Doutorado e estágios Pós-Doutorais, e na Resolução CONSUNI 02/2014 [CONSUNI 2014], que regulamenta o Programa de Capacitação e o Plano Anual de Capacitação dos TAEs. Em consonância com a Resolução CEPEC 1286/2014, o INF instrui o afastamento de docentes com a Resolução CD/INF n. 01 de 2014, que dá suporte ao planejamento administrativo e incentiva a participação de seus docentes, em cursos de doutorado, pós-doutorado e capacitação, no país e no exterior, de acordo com a sua política de pessoal para o ensino, a pesquisa, a extensão e a administração.

Por fim, cabe ao NDE monitorar o PPC e, consequentemente, promover ações que promovam a qualidade dos serviços oferecidos tanto por docentes quanto por TAEs.

Requisitos legais e normativos

O curso de Bacharelado em Engenharia de Software contempla requisitos legais específicos, a saber, Libras, Política de Educação Ambiental, e Educação das Relações Étnico-raciais e para o Ensino da História e Cultura Afro-brasileira e Indígena, conforme detalhado nas seções seguintes.

A atenção a tais requisitos se verifica por meio de disciplinas e de um Programa de Extensão, devidamente institucionalizado na unidade executora do curso. Está além do escopo do presente documento detalhar o Programa de Extensão. Abaixo, apenas aspectos relevantes para o PPC são destacados.

Libras

O Decreto 5.626/2005 regulamenta a Lei 10.436, de 24 de abril de 2002, e o artigo 18 da Lei 10.098, de 19 de dezembro de 2000. Segundo esse Decreto, a disciplina curricular Libras é obrigatória para vários cursos, dentre eles, as licenciaturas e os cursos de Fonoaudiologia. O Decreto também estabelece que, nos demais cursos, a disciplina curricular Libras seja optativa, conforme o Capítulo II, § 2º: "a Libras constituir-se-á em disciplina curricular optativa nos demais cursos de educação superior".

Em atenção a tal requisito legal, o Bacharelado em Engenharia de Software inclui a disciplina Libras como optativa, no oitavo período do curso.

Política de educação ambiental

A Lei 9.795, de 27 de abril de 1999, institui a Política Nacional de Educação Ambiental, que é regulamentada pelo Decreto 4.281, de 25 de junho de 2002. Conforme essa Lei, Seção II, Art. 10, "a educação ambiental será desenvolvida como uma prática educativa integrada, contínua e permanente em todos os níveis e modalidades do ensino formal". Adicionalmente, lê-se na Seção II, Art. 10, § 1º: "a educação ambiental **não** deve ser implantada como disciplina específica no currículo de ensino". O destaque (negrito) é do presente documento.

O Bacharelado em Engenharia de Software trata a Educação Ambiental por duas linhas contínuas de atuação: uma delas baseada no planejamento das

disciplinas e outra em programa institucional do Instituto de Informática, ambas comentadas abaixo.

Planejamento das disciplinas

O planejamento de cada turma deve se inspirar em possíveis estratégias de inserção de questões ambientais. Por exemplo, adoção de material em formato digital em vez de formato impresso.

A apresentação do conteúdo de cada disciplina também pode se beneficiar do volume significativo de dados sobre o meio ambiente. Por exemplo, área desmatada ao longo do tempo; áreas de preservação; consumo de energia; emissão de CO_2; qualidade do ar; consumo de materiais poluentes e geração de lixo eletrônico. Tais dados podem ser empregados para ilustrar o funcionamento de algoritmos e visualização de informações, dentre outras possibilidades.

Convém destacar que esses exemplos devem ser vistos como elementos de inspiração, a serem renovados continuamente.

Programa institucional

A UFG executa um Plano de Logística Sustentável (PLS), que estabelece práticas de sustentabilidade e racionalização de gastos e processos na administração pública [PLS]. Esse plano é internalizado no Instituto de Informática por um Programa Institucional de extensão.

O PLS une o cotidiano da prática acadêmica com atitudes “sustentáveis” por meio de recomendações simples, como a impressão em ambos os lados de uma folha e a redução do uso de copos descartáveis, dentre muitas outras.

O Programa Institucional reúne ações que contemplam as orientações do PLS. O objetivo é colocar em prática essas orientações. Por exemplo, enquanto o PLS sugere a coleta seletiva, esse programa cria um repositório para coleta de pilhas e baterias já utilizadas, além de assegurar que aquele material coletado será descartado de forma correta.

O Programa Institucional possui objetivos e ações que incluem a educação ambiental. As opções de ações variam. Dentre elas uma é constante: avaliação dos resultados. Dentre as demais:

- Cursos de extensão cuja inscrição seja lixo eletrônico;
- Elaboração de material de conscientização sobre consumo parcimonioso de água e energia;
- Palestras e cursos sobre TI Verde (*green computing*);
- Pesquisa sobre consumo de energia por *datacenters*;
- Divulgação e destaque de informações sobre o meio ambiente;
- Monitoramento e divulgação de informações ambientes sobre Goiás;
- Divulgação de softwares que promovem o meio ambiente, por exemplo, evitam deslocamentos, evitam consumo de energia.

Resolução Nº 1 de 17 de junho de 2004

A Resolução Nº 1, de 17 de junho de 2004, trata das Diretrizes Curriculares Nacionais para a Educação das Relações Étnico-Raciais e para o Ensino de História e Cultura Afro-Brasileira e Africana.

O objetivo é claro: "combater o racismo e as discriminações que atingem particularmente os negros. Nessa perspectiva, propõe a divulgação e produção de conhecimentos, a formação de atitudes, posturas e valores que eduquem cidadãos orgulhosos de seu pertencimento étnico-racial, descendentes de africanos, povos indígenas, descendentes de europeus, de asiáticos. "

Tal objetivo pressupõe a "adoção de políticas educacionais e de estratégias pedagógicas de valorização da diversidade", conforme consta na Resolução, assim como os princípios a serem observados para atendê-la: (a) consciência política e histórica da diversidade; (b) fortalecimento de identidades e de direitos e (c) ações educativas de combate ao racismo e a discriminações.

O Art. 7.º da Resolução ainda destaca: "as instituições de ensino superior, respeitada a autonomia que lhe é devida, incluirão nos conteúdos de

disciplinas e atividades curriculares dos diferentes cursos que ministram, a Educação das Relações Étnico-Raciais, bem como o tratamento de questões e temáticas que dizem respeito aos afrodescendentes. ”

Tendo em vista o requisito legal estabelecido pela Resolução, duas linhas de atuação são adotadas pelo Bacharelado em Engenharia de Software: inserção de conteúdo pertinente em duas disciplinas curriculares obrigatórias e o Programa Institucional de extensão do Instituto de Informática. Ambas comentadas abaixo.

Disciplinas de graduação obrigatórias

As disciplinas “Computador e Sociedade” (p. 45) e “Interação Humano-Computador” (p. 51) incluem, em seus ementários, tópicos pertinentes às exigências da Resolução. Em particular, remetem para o conhecimento de questões pertinentes ao continente africano e para o conhecimento e respeito à diversidade.

Programa institucional

O Programa Institucional de extensão do Instituto de Informática será realizado por meio de ações, boa parte com ênfase na socialização de conhecimento sobre questões pertinentes à formação dos brasileiros, o que invariavelmente inclui os negros e indígenas. Esse conhecimento é indispensável para a promoção da diversidade, do respeito às diferenças e da igualdade independente das crenças, do sexo, da idade, da cor, da condição social.

O Instituto de Informática, por meio desse programa de extensão, dedicado exclusivamente às relações étnico-raciais e ao meio ambiente, oferece atuação contínua sobre esses tópicos, durante toda a permanência dos seus estudantes nessa unidade, sejam de graduação ou pós-graduação.

Especificamente sobre questões de cunho étnico-racial, sem a intenção de ser uma apresentação exaustiva, nem restritiva, são identificadas algumas ações possíveis:

- Ações afirmativas por meio de cursos de extensão.
- Ações para promoção da diversidade.

- Palestras sobre a construção de uma sociedade justa.
- Palestras sobre a diversidade da formação dos brasileiros.
- Palestras sobre a história afro-brasileira, sobre a história africana.
- Palestras sobre os povos indígenas.
- Apresentações artísticas que valorizem a cultura africana e indígena.

Diretrizes nacionais para Educação em Direitos Humanos

A disciplina "Computador e Sociedade" (p. 45) trata questões relevantes acerca de direitos humanos: aspectos profissionais; participação de mulheres na computação e evolução de aplicações com acessibilidade, dentre outras, nas quais os direitos humanos precisam de discussões mais aprofundadas na computação. Na disciplina "Interação Humano-Computador" (p. 51) as características humanísticas, culturais, de direitos humanos e de aspectos políticos serão abordados em mais profundidade.

Proteção dos direitos da pessoa com Transtornos do Espectro Autista

A proteção dos direitos da pessoa com transtorno do espectro autista é fundamentada na Lei N.º 12.764 de 27 de dezembro de 2012, que institui a política nacional de proteção dos direitos da pessoa com transtorno do espectro autista, e altera o § 3.º do Art. 98 da Lei N.º 8.112, de 11 de dezembro de 1990. Esse requisito legal é atendido por meio da disciplina obrigatória "Interação Humano-Computador", que inclui em seu ementário tópicos pertinentes ao tratamento de características humanísticas e biológicas na construção de interfaces de usuário, o que possibilita a socializar informações relevantes para que portadores do Transtorno do Espectro Autista possam ser adequadamente considerados por meio de suas necessidades especiais.

Estrutura Curricular

Matriz curricular

A matriz curricular do curso perfaz 44 disciplinas descritas nas Tabelas 1, 2 e 3. A Tabela 1 apresenta a lista das 17 disciplinas de formação básica em computação. Em particular, são 4 disciplinas oferecidas pelo Instituto de Matemática e Estatística (IME) e as outras 13 oferecidas pelo INF. Todas elas são obrigatórias e pertencem ao Núcleo Comum (NC).

Tabela 1 Disciplinas de formação básica (obrigatórias). Não há co-requisito.

Nº	Disciplina	Pré	Unidade	TEO	PRA	CHT	Núcleo
1	Computação e Sociedade		INF	32	0	32	NC
2	Introdução à Programação		INF	48	80	128	NC
3	Cálculo 1A		IME	96	0	96	NC
4	Fundamentos de Matemática para Computação		INF	64	0	64	NC
5	Arquitetura de Computadores	4	INF	64	0	64	NC
6	Algoritmos e Estruturas de Dados 1	2	INF	32	32	64	NC
7	Probabilidade e Estatística A	3	IME	64	0	64	NC
8	Álgebra Linear		IME	64	0	64	NC
9	Lógica Matemática	4	INF	64	0	64	NC
10	Programação Orientada a Objetos	2	INF	32	32	64	NC
11	Algoritmos e Estruturas de Dados 2	6	INF	64	0	64	NC
12	Linguagens e Paradigmas de Programação	3	INF	32	32	64	NC
13	Engenharia de Software		INF	64	0	64	NC
14	Análise e Projeto de Algoritmos	4, 11	INF	64	0	64	NC
15	Interação Humano-Computador	6, 7	INF	32	32	64	NC
16	Banco de Dados	9	INF	48	16	64	NC
17	Projeto de Software	10, 13	INF	64	0	64	NC

As disciplinas da Tabela 1 fornecem uma sólida formação em Computação, o que inclui o recomendado para a formação em Matemática. Além dessas, outras seis de Computação são oferecidas como optativas na Tabela 3 (p. 37).

As disciplinas da Tabela 2, por outro lado, são específicas da Engenharia de Software. Todas elas são obrigatórias, fazem parte do Núcleo Específico (NE) e são oferecidas pelo INF.

Tabela 2 Disciplinas de Engenharia de Software (obrigatórias). Não há pré(co)-requisito.

Nº	Disciplina	Pré	Unidade	TEO	PRA	CHT	Núcleo
18	Construção de Software		INF	48	80	128	NE
19	Modelagem de Software		INF	16	48	64	NE
20	Processos de Software		INF	32	32	64	NE
21	Engenharia de Sistemas		INF	64	0	64	NE
22	Design de Software		INF	48	80	128	NE
23	Domínios de Software		INF	16	48	64	NE
24	Processos de Qualidade de Software		INF	48	16	64	NE
25	Gerência de Projeto de Software		INF	32	32	64	NE
26	Software Concorrente e Distribuído		INF	32	32	64	NE
27	Experiência do Usuário de Software		INF	32	32	64	NE
28	Arquitetura de Software		INF	32	32	64	NE
29	Requisitos de Software		INF	32	32	64	NE
30	Governança e Gestão de Serviços de Software		INF	48	16	64	NE
31	Software para Sistemas Ubíquos		INF	32	32	64	NE
32	Software para Persistência de Dados		INF	32	32	64	NE
33	Padrões de Arquitetura de Software		INF	32	32	64	NE
34	Testes de Software		INF	32	32	64	NE
35	Prática em Engenharia de Software		INF	0	320	320	NE

São nove as disciplinas optativas. O estudante terá que escolher três delas, conforme as opções oferecidas na Tabela 3 abaixo.

Tabela 3 Disciplinas optativas. Não há co-requisito.

Nº	Disciplina (opção)	Pré	Unidade	TEO	PRA	CHT	Núcleo	Optativa
36	Pesquisa Operacional	8	INF	64	0	64	NE	1
37	Linguagens Formais e Autômatos	4	INF	64	0	64	NE	
38	Sistemas Operacionais	5	INF	64	0	64	NE	
39	Compiladores	11, 37	INF	64	0	64	NE	2
40	Redes de Computadores		INF	64	0	64	NE	
41	Introdução à Língua Brasileira de Sinais		LETRAS	64	0	64	NE	
42	Sistemas Distribuídos	10	INF	64	0	64	NE	3
43	Metodologia e Experimentação em Engenharia de Software		INF	64	0	64	NE	
44	Mercado e Economia de Software		INF	64	0	64	NE	

Núcleo livre

O Núcleo Livre (NL) é o conjunto de conteúdos voltado para promover a interdisciplinaridade e a transdisciplinaridade, dentre outros objetivos.

A carga horária definida pelo BES para o NL é de 128 horas, que é o mínimo estabelecido pelo Regimento Geral dos Cursos de Graduação (RGCG) da UFG [RGCG]. Tais 128 horas perfazem 4% da carga horária do curso, conforme a Tabela 4 (p. 38).

Atividades complementares

O estudante do BES deve cumprir o mínimo de 192 horas em atividades complementares para a integralização curricular. Essas horas correspondem a 6% da carga horária do curso, conforme a distribuição da carga horária do curso na Tabela 4 (p. 38).

As atividades complementares contemplam ações de monitoria, produção científica, extensão, produção técnica e de representação e qualificação, dentre outras. O cumprimento das atividades complementares é estabelecido formalmente pelo *Regulamento de Atividades Complementares do Bacharelado em Engenharia de Software.*

Distribuição da carga horária

As Tabelas 1, 2 e 3, juntamente com a definição das horas do Núcleo Livre (NL) e das atividades complementares, seções acima, resultam nos valores compilados na Tabela 4.

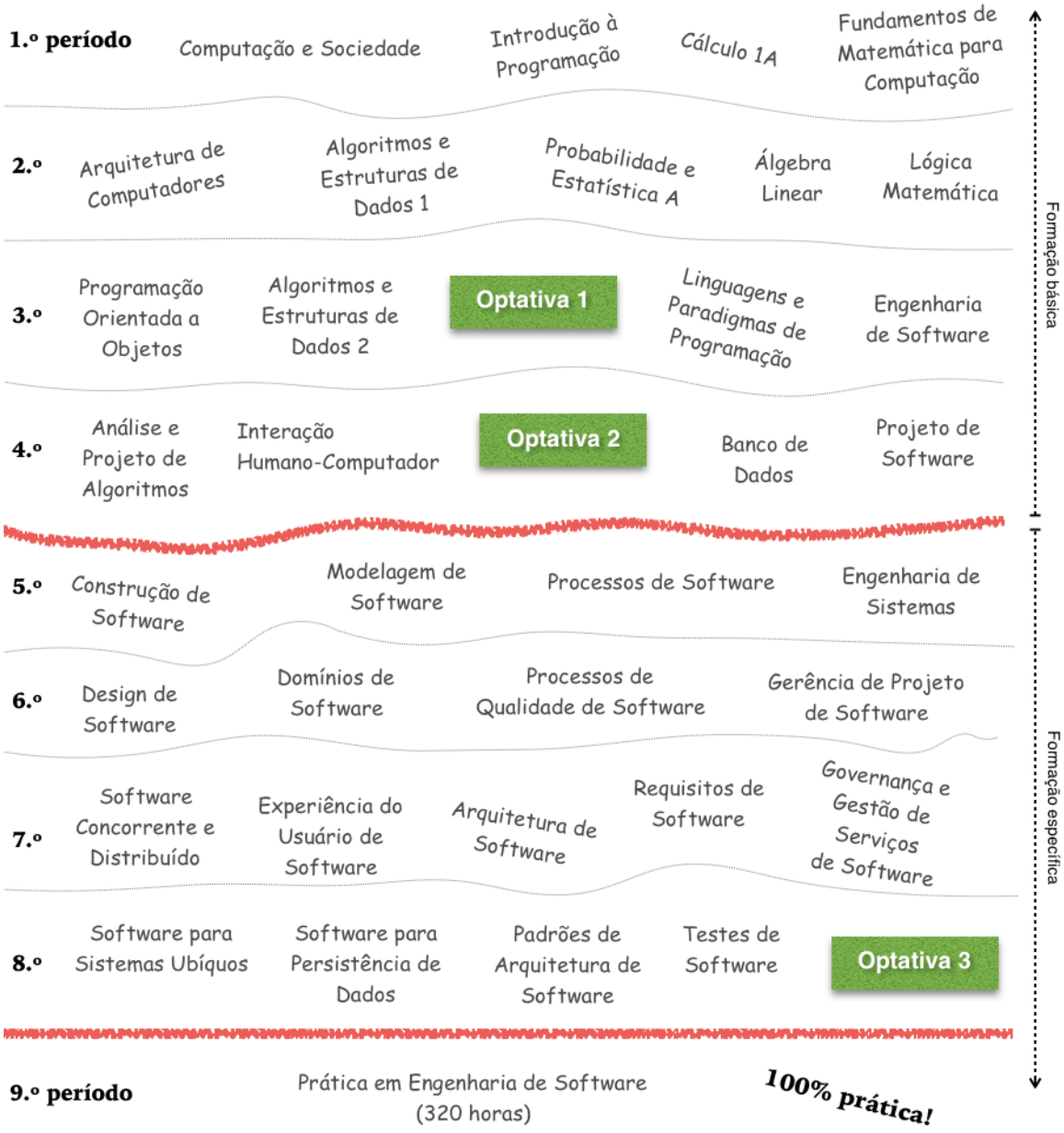
Tabela 4 Distribuição da carga horária.

Componentes curriculares	CHT	Percentual
Núcleo Comum (NC)	1088	34%
Núcleo Específico (NE) (obrigatório)	1600	50%
Núcleo Específico (NE) (optativo)	192	6%
Núcleo Livre (NL)	128	4%
Atividades Complementares	192	6%
CARGA HORÁRIA TOTAL (CHT)	**3200**	**100%**

Fluxo sugerido

Conforme o diagrama abaixo, as disciplinas estão organizadas em nove períodos letivos. Os quatro primeiros períodos concentram as disciplinas de formação básica em Computação: disciplinas de Computação propriamente ditas e aquelas de Matemática. Os demais períodos incluem disciplinas com ênfase na formação específica em Engenharia de Software.

Figura 2 Fluxo sugerido.



Os nove semestres são detalhados nas tabelas seguintes.

1º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Computação e Sociedade	32	Obrigatória	NC
Introdução à Programação	128	Obrigatória	NC
Cálculo 1A	96	Obrigatória	NC
Fundamentos de Matemática para Computação	64	Obrigatória	NC
Carga horária do período	320		
Carga horária acumulada	320		

2º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Arquitetura de Computadores	64	Obrigatória	NC
Algoritmos e Estruturas de Dados 1	64	Obrigatória	NC
Probabilidade e Estatística A	64	Obrigatória	NC
Álgebra Linear	64	Obrigatória	NC
Lógica Matemática	64	Obrigatória	NC
Carga horária do período	320		
Carga horária acumulada	640		

3º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Programação Orientada a Objetos	64	Obrigatória	NC
Algoritmos e Estruturas de Dados 2	64	Obrigatória	NC
Linguagens e Paradigmas de Programação	64	Obrigatória	NC
Engenharia de Software	64	Obrigatória	NC
OPTATIVA 1	64	Optativa	NE
Carga horária do período	320		
Carga horária acumulada	960		

4º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Análise e Projeto de Algoritmos	64	Obrigatória	NC
Interação Humano-Computador	64	Obrigatória	NC
Banco de Dados	64	Obrigatória	NC
Projeto de Software	64	Obrigatória	NC
OPTATIVA 2	64	Optativa	NE
Carga horária do período	320		
Carga horária acumulada	1280		

5º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Construção de Software	128	Obrigatória	NE
Modelagem de Software	64	Obrigatória	NE
Processos de Software	64	Obrigatória	NE
Engenharia de Sistemas	64	Obrigatória	NE
Carga horária do período	320		
Carga horária acumulada	1600		

6º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Design de Software	128	Obrigatória	NE
Domínios de Software	64	Obrigatória	NE
Processos de Qualidade de Software	64	Obrigatória	NE
Gerência de Projeto de Software	64	Obrigatória	NE
Carga horária do período	320		
Carga horária acumulada	1920		

7º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Software Concorrente e Distribuído	64	Obrigatória	NE
Experiência do Usuário de Software	64	Obrigatória	NE
Arquitetura de Software	64	Obrigatória	NE
Requisitos de Software	64	Obrigatória	NE
Governança e Gestão de Serviços de Software	64	Obrigatória	NE
Carga horária do período	320		
Carga horária acumulada	2240		

8º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Software para Computação Ubíqua	64	Obrigatória	NE
Software para Persistência de Dados	64	Obrigatória	NE
Padrões de Arquitetura de Software	64	Obrigatória	NE
Testes de Software	64	Obrigatória	NE
OPTATIVA 3	64	Optativa	NE
Carga horária do período	320		
Carga horária acumulada	2560		

9º PERÍODO			
Disciplina	**CHT**	**Natureza**	**Núcleo**
Prática em Engenharia de Software	320	Obrigatória	NE
Carga horária do período	320		
Carga horária acumulada	2880		

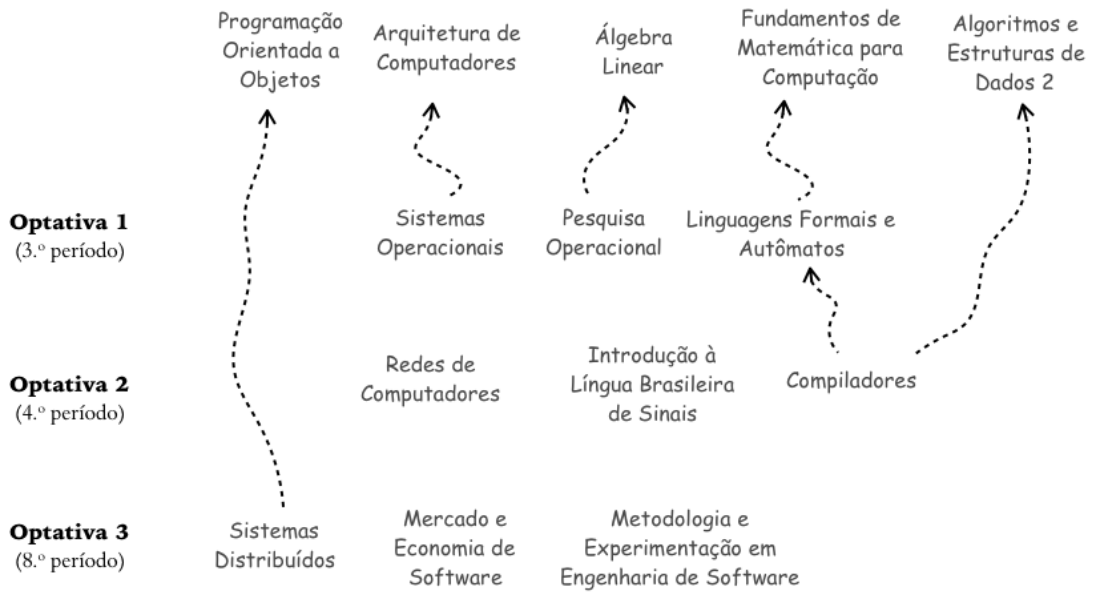
Pré-requisitos

Existem pré-requisitos entre as disciplinas. Aqueles das disciplinas optativas são exibidos na Figura 3, enquanto aqueles das disciplinas obrigatórias na Figura 4 (p. 44).

Observe que as disciplinas optativas estão agrupadas na Figura 3 conforme as opções oferecidas aos estudantes. Por exemplo, a disciplina OPTATIVA 1, sugerida para ser cursada no 3.º período, será escolhida pelo estudante dentre as seguintes opções: "Sistemas Operacionais" (p. 53), "Pesquisa Operacional" (p. 53) ou "Linguagens Formais e Autômatos" (p. 53). Para a disciplina OPTATIVA 2, sugerida para ser cursada no 4.º período, as opções são: "Redes de Computadores" (p. 54), "Introdução à Língua Brasileira de Sinais" (p. 53) e "Compiladores" (p. 54). Para a disciplina OPTATIVA 3, sugerida para ser cursada no 8.º período, as opções são: "Sistemas Distribuídos" (p. 55), "Mercado e Economia de Software" (p. 74) e "Metodologia e Experimentação em Engenharia de Software" (p. 74).

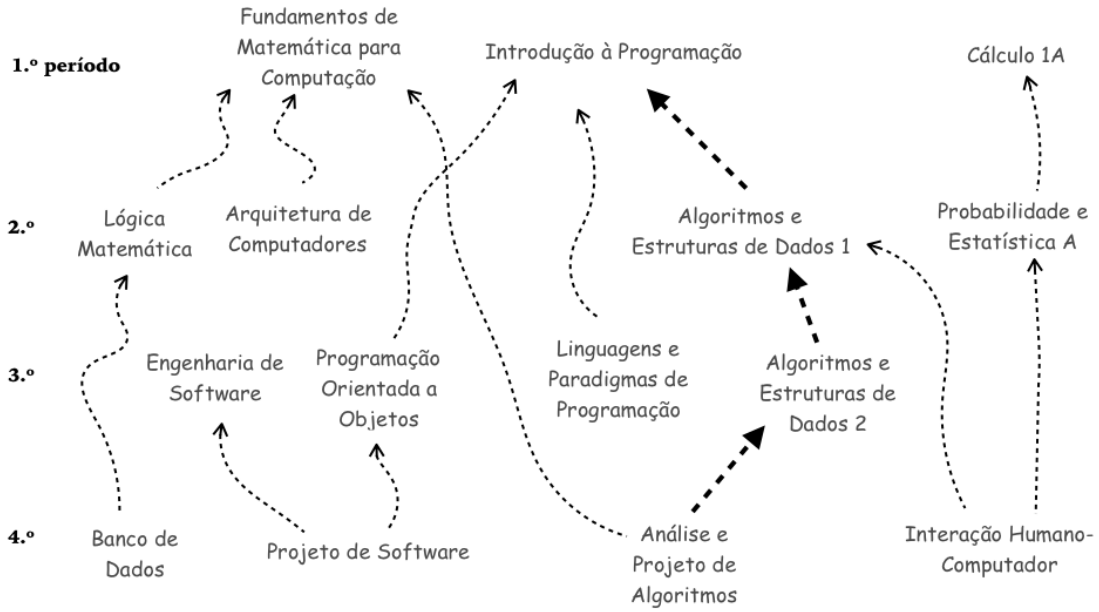
Ainda convém destacar que algumas disciplinas optativas dependem de outras obrigatórias. Por exemplo, conforme ilustrado abaixo, "Pesquisa Operacional" tem como pré-requisito a disciplina obrigatória "Álgebra Linear".

Figura 3 Pré-requisitos das disciplinas optativas.



Todos os pré-requisitos das disciplinas obrigatórias são exibidos na Figura 4. As disciplinas seguem alinhadas, na horizontal, do primeiro até o último (nono) período do curso. Em particular, o caminho mais longo, não é o caminho crítico, encontra-se devidamente destacado dos demais.

Figura 4 Pré-requisitos das disciplinas obrigatórias.



Disciplinas de formação básica (obrigatórias)

Computação e Sociedade

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	0	32	1.º	INF	Sim	Nenhum

Ementa

1. História da computação.
2. Estudo e análise de casos de aplicação de computadores na sociedade e para o meio ambiente.
3. Subáreas da computação e áreas interdisciplinares.
4. Importância e desafios da computação no Brasil e no mundo.
5. Cursos de computação e aspectos profissionais: tipos de cursos, perfis profissionais, demanda do mercado, organizações e associações na área, regulamentação da profissão.
6. Leis e normas relacionadas à Informática.
7. Questões ambientais, raciais, de saúde e de inclusão digital relacionadas à Computação.
8. Ética na Computação.
9. Empresas de tecnologia da informação.
10. Incubadoras de empresas.

Introdução à Programação

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	48	80	128	1.º	INF	Sim	Nenhum

Ementa

1. Introdução a algoritmos.
2. Conceitos básicos de programas: constantes; tipos de dados primitivos; variáveis; atribuição; entrada e saída de dados; expressões; estruturas de decisão; estruturas de repetição.
3. Ponteiro.
4. Estruturas de dados homogêneas e heterogêneas: vetores, matrizes, cadeias de caracteres, registros. Subprogramas: funções; passagens de parâmetros por valor e por referência, recursividade.
5. Manipulação de arquivos: abertura, fechamento, leitura e gravação.

6. Tipos de acesso a arquivos: sequencial e indexado.
7. Tipos de arquivos (texto e binário).
8. Transcrição de algoritmos para uma linguagem de programação.
9. Domínio de uma linguagem de programação: sintaxe e semântica; interpretação e compilação de programas; ambiente de desenvolvimento de programas; estilo de codificação; documentação de código; técnicas de depuração e técnicas de *profiling*; desenvolvimento e uso de bibliotecas.

Cálculo 1A

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	96	0	96	1.º	IME	Sim	Nenhum

Ementa

1. Números reais. Funções reais de uma variável real e suas inversas.
2. Noções sobre cônicas.
3. Limite e continuidade.
4. Derivadas e aplicações.
5. Polinômio de Taylor.
6. Integrais. Técnicas de Integração. Integrais impróprias.
7. Aplicações.

Fundamentos de Matemática para Computação

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	64	0	64	1.º	INF	Sim	Nenhum

Ementa

1. Noções de lógica.
2. Introdução a demonstrações.
3. Indução matemática, Recursividade e Relações de Recorrência, Conjuntos e Combinatória.
4. Séries e sequências.
5. Relações e Funções.
6. Representações numéricas e Mudança de base numérica.

Arquitetura de Computadores

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	48	16	64	2.º	INF	Sim	Fundamentos de Matemática para Computação

Ementa

1. Visão geral dos computadores modernos.
2. Evolução da arquitetura dos computadores.
3. Memória e representação de dados e instruções.
4. Processador, ciclo de instrução, formato e endereçamento.
5. Noções básicas de programação em linguagem de montagem.
6. Dispositivos de entrada e saída.
7. Sistemas de interconexão (barramentos).
8. Interfaceamento e técnicas de entrada e saída.
9. Hierarquia de memória.
10. Introdução a arquiteturas paralelas e métricas de desempenho.

Algoritmos e Estruturas de Dados 1

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	2.º	INF	Sim	Introdução à Programação

Ementa

1. Noções de complexidade de algoritmos (notações de complexidade).
2. Algoritmos de pesquisa: pesquisa sequencial e binária.
3. Algoritmos de ordenação.
4. Tipos abstratos de dados.
5. Estruturas de dados utilizando vetores: pilhas, filas, listas (simples e circulares).
6. Estruturas de dados com alocação dinâmica de memória: pilhas, filas, listas (simplesmente encadeadas, duplamente encadeadas e circulares).

Probabilidade e Estatística A

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	64	0	64	2.º	IME	Sim	Cálculo 1A

Ementa

1. Estatística descritiva. Noções sobre amostragem.
2. Introdução à teoria de conjuntos.

3. Introdução à teoria de probabilidade: espaço amostral, eventos, frequência relativa, fundamentos de probabilidade, probabilidade condicional, eventos independentes e teorema de Bayes.
4. Variáveis aleatórias: conceitos básicos, esperança e variância.
5. Distribuições discretas de probabilidade: uniforme, binomial e Poisson.
6. Distribuições contínuas de probabilidade: uniforme, exponencial, normal e t-Student.
7. Estimação pontual e intervalar para uma população: média e proporção.
8. Teste de hipóteses para uma população: média e proporção.
9. Correlação linear e regressão linear simples.

Álgebra Linear

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	64	0	64	2.º	IME	Sim	Nenhum

Ementa

1. Sistemas lineares e matrizes.
2. Espaços vetoriais.
3. Transformações lineares.
4. Autovalores e autovetores.
5. Espaços com produto interno.

Lógica Matemática

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	64	0	64	2.º	IME	Sim	Fundamentos de Matemática para Computação

Ementa

1. Noções básicas: linguagem natural vs linguagens formais; verdade, validade, satisfatibilidade; lógica proposicional (sintaxe e semântica, propriedades e relações semânticas, consequência lógica, simplificação de fórmulas); lógica de primeira ordem (sintaxe e semântica, propriedades e relações semânticas, formas normais);
2. Métodos de validação: métodos diretos de prova; métodos de prova por contradição; indução estrutural.
3. Linguagem para experimentação.
4. Aplicações básicas.

Programação Orientada a Objetos

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	3.º	INF	Sim	Introdução à Programação

Ementa

1. Abstração e tipos abstratos de dados.
2. Classes, métodos, encapsulamento, interface. Mensagens, instâncias e inicialização. Herança e composição. Polimorfismo.
3. Uso de uma linguagem orientada a objetos.
4. Noções de UML.
5. Noções de padrões de projeto orientado a objetos.

Algoritmos e Estruturas de Dados 2

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	64	0	64	3.º	INF	Sim	Algoritmos e Estruturas de Dados 1

Ementa

1. Árvores: formas de representação, recursão em árvores, árvores binárias, árvores binárias de busca, árvores balanceadas (AVL e rubro-negras).
2. Filas de prioridades. *Heaps*, *Heapsort*. *Hashing*: tipos de funções de *hashing*; tratamento de colisões.
3. Definições de Grafos.
4. Estruturas de Dados para representação de grafos.
5. Algoritmos básicos em grafos.

Linguagens e Paradigmas de Programação

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	3.º	INF	Sim	Introdução à Programação

Ementa

1. Estudo dos conceitos de linguagens de programação e dos paradigmas de programação: procedural, funcional, lógico, orientado a objeto e *script*.
2. Reflexão sobre as características desejáveis em uma linguagem de programação e os critérios de seleção de linguagens de acordo com as especificidades dos domínios de aplicação.

3. Descrição de sintaxe e semântica.
4. Estudo sobre tipos de dados, estruturas de controle, ambientes de execução, variáveis, expressões e subprogramas em linguagens de programação.

Engenharia de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	64	0	64	3.º	INF	Sim	Nenhum

Ementa

1. Requisitos de software.
2. Projeto (design) de software.
3. Construção de software.
4. Teste de software.
5. Manutenção de software.
6. Gerência de configuração de software.
7. Gerência de projeto de software.
8. Processo de engenharia de software.
9. Modelos e métodos de engenharia de software.
10. Qualidade de software.
11. Prática profissional de engenharia de software.
12. Economia para engenharia de software.
13. Fundamentos de engenharia.

Análise e Projeto de Algoritmos

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisitos:
NC	64	0	64	4.º	INF	Sim	Fundamentos de Matemática para Computação Algoritmos e Estruturas de Dados 2

Ementa

1. Medidas de complexidade, análise assintótica de limites de complexidade para algoritmos iterativos e recursivos, técnicas de prova de cotas inferiores.
2. Corretude de Algoritmos.
3. Exemplos de análise de algoritmos.
4. Técnicas de projeto de algoritmos: dividir para conquistar, programação dinâmica, algoritmos gulosos.

5. Introdução à NP-Completude.

Interação Humano-Computador

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisitos:
NC	32	32	64	4.º	INF	Sim	Probabilidade e Estatística A Algoritmos e Estruturas de Dados 1

Ementa

1. Aspectos gerais sobre interação humano-computador.
2. Características humanísticas e biológicas, envolvendo questões sobre genealogia, gênero, aspectos étnicos, raciais e culturais, direitos e aspectos políticos, deficiências, limitações e capacidades, dentre outros.
3. Ciclo da interação e principais problemas. Metas de usabilidade e experiência do usuário.
4. Fatores humanos em software interativo: teoria, princípios e regras básicas. Modelos conceituais e metáforas.
5. Estilos de interação. Elementos de interação (menus, formulários, manipulação direta e outros). Voz, linguagem natural, sons, páginas Web. Padrões para interface.
6. Localização e internacionalização.
7. Princípios de projeto de interfaces humano-computador. Métodos de projeto de interação. Projeto visual (cores, ícones, fontes e outros). Tempo de resposta e retroalimentação.
8. Dispositivos de interação.
9. Métodos de avaliação de interfaces: avaliação heurística, abordagens para testes realizados com apoio de usuários, técnicas de testes para páginas Web, entre outros.
10. Visão geral de ferramentas de desenvolvimento de interfaces humano-computador.

Banco de Dados

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisitos:
NC	48	16	64	4.º	INF	Sim	Lógica Matemática

Ementa

1. Conceitos básicos de Banco de Dados.

2. Modelo relacional.
3. Linguagens para Banco de Dados: álgebra relacional, cálculo relacional e SQL.
4. Modelo Entidade-Relacionamento e extensões.
5. Mapeamento ER-relacional.
6. Normalização.

Projeto de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisitos:
NC	64	0	64	4.º	INF	Sim	Engenharia de Software Programação Orientada a Objetos

Ementa

1. Fundamentos de *design* de software.
2. Questões básicas de *design* de software: concorrência, controle e tratamento de eventos, persistência de dados, distribuição, tratamento de erro e exceção, tolerância a falhas, interação e apresentação, e segurança. Estrutura e arquitetura de software.
3. Projeto de interface de usuário.
4. Análise de qualidade e avaliação de *design* de software.
5. Notação de *design* de software.
6. Métodos e estratégias de *design* de software.
7. Ferramentas de *design* de software.

Disciplinas optativas

Além das disciplinas definidas nessa seção, duas outras são optativas e específicas de Engenharia de Software: “Mercado e Economia de Software” (p. 74) e “Metodologia e Experimentação em Engenharia de Software” (p. 74).

Sistemas Operacionais

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NE	32	32	64	3.º	INF	Não	Arquitetura de Computadores

Ementa

1. Conceitos de Hardware e Software.
2. Tipos de Sistemas Operacionais. Organização da Estrutura Interna do Sistema Operacional. Gerência de Processos. Gerência do Processador. Gerência de Memória. Gerência de Dispositivos de Entrada e Saída. Sistemas de Arquivos.
3. Estudos de casos de sistemas operacionais atuais.

Pesquisa Operacional

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NE	64	0	64	3.º	INF	Não	Álgebra Linear

Ementa

1. Modelagem.
2. Problema de Programação Linear (PL). Resolução gráfica de PL.
3. Algoritmo Simplex. Dualidade. Algoritmo Simplex-Dual.
4. Pós-otimização e Análise de Sensibilidade.

Linguagens Formais e Autômatos

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NE	64	0	64	3.º	INF	Não	Fundamentos de Matemática para Computação

Ementa

1. Conceitos básicos de linguagens.
2. Mecanismos geradores (gramáticas) e reconhecedores (determinísticos e não determinísticos) de linguagens regulares, livres de contexto e

sensíveis ao contexto; relação entre estas classes de linguagens e suas principais propriedades.
3. Hierarquia de Chomsky.

Redes de computadores

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NE	32	32	64	4.º	INF	Não	Sistemas Operacionais

Ementa

1. Fundamentos.
2. Arquitetura de Redes TCP/IP (Internet).
3. Camadas de Aplicação, Transporte, Rede, Enlace e Física.
4. Gerenciamento de Redes.
5. Redes Sem Fio e Mobilidade.
6. Estudo de Caso de Tópicos Emergentes em Redes.

Introdução à Língua Brasileira de Sinais

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NE	64	0	64	4.º	LETRAS	Não	Nenhum

Ementa

1. Introdução às práticas de compreensão e produção em LIBRAS através do uso de estruturas e funções comunicativas elementares.
2. Concepções sobre a Língua de Sinais.
3. O surdo e a sociedade.

Compiladores

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NE	64	0	64	4.º	INF	Não	Linguagens Formais e Autômatos Algoritmos e Estruturas de Dados 2

Ementa

1. A estrutura de um compilador.
2. Análises léxica e sintática.
3. Definições dirigidas por sintaxe e análise semântica.
4. Organização da tabela de símbolos.
5. Representação intermediária do programa-fonte.

6. Geração do código-objeto da compilação.
7. Introdução à otimização do código-objeto.
8. Implementação de um compilador.

Sistemas Distribuídos

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NE	32	32	64	8.º	INF	Não	Programação Orientada a Objetos

Ementa

1. Introdução a Sistemas Distribuídos.
2. Invocação Remota.
3. Objetos Distribuídos.
4. Arquiteturas Orientadas a Serviços e utilização de serviços Web.
5. Computação Móvel e Ubíqua. Estudo de Casos de Tópicos Emergentes em Sistemas Distribuídos.

Disciplinas específicas de Engenharia de Software

Em geral, disciplinas são apresentadas em um padrão composto por ementa, bibliografia básica e bibliografia complementar. As disciplinas de Engenharia de Software, descritas nessa seção, acrescentam uma dimensão denominada "condições mínimas", elucidada a seguir.

Condições mínimas

Idealmente uma disciplina deve ser não ambígua e verificável. Por não ambígua entenda que, sem interferir na autonomia metodológica do docente, o registro de cada disciplina deve ser preciso no foco e no compromisso dela perante as demais ações do curso. Por verificável entenda que o processo de avaliação dos estudantes deve ser bem definido.

O esforço para descrever uma disciplina sem ambiguidades e verificável não implica em perfeição. Em vez desse alvo, o objetivo é reduzir as variações constatadas e decorrentes de quem ministra a disciplina. Essas variações resultam em flutuação da qualidade, cuja redução é, de fato, o que justifica o cuidado com a forma empregada para especificar as disciplinas.

A ementa de uma disciplina define o escopo do discurso da disciplina, ou seja, o que deve ser abordado na disciplina. Para cada tópico da ementa uma carga horária é definida. Embora todos os tópicos sejam relevantes, alguns devem receber maior atenção do que outros. O objetivo é zelar pela consistência e coerência do projeto pedagógico como um todo. Noutras palavras, a simples inversão ou alteração dessas horas só faz sentido se a repercussão for considerada em todo o conjunto das disciplinas, o que resultaria em projeto pedagógico distinto. Adicionalmente, esta atribuição de carga horária por tópico não é arbitrária, mas definida de forma criteriosa para atender o perfil definido para o egresso do curso.

As condições mínimas definem o quanto o escopo, definido pela ementa, deve ser do domínio do estudante para aprovação na disciplina em questão. Ou seja, enquanto a ementa define o objeto, as condições mínimas definem o mínimo que se espera do estudante em relação a tal objeto para que ele seja aprovado na disciplina. Essas condições são identificadas por um conjunto de atividades,

onde cada atividade é acompanhada de um nível. Ou seja, as atividades definem o que o estudante deve estar apto a fazer, a realizar, acerca do objeto em questão, e o nível define a proficiência esperada ao realizar a atividade em questão. São três os níveis:

- **Segue instruções**. O estudante é capaz de realizar passos. O estudante é capaz de executar as instruções requisitadas pelo docente. Por exemplo, o estudante deve ser capaz de "compilar código fonte" usando a ferramenta especificada.
- **Faz com orientação**. O estudante faz com orientação quando é capaz de realizar uma atividade, sem que instruções ou passos detalhados sejam fornecidos, contudo, depende da assistência do docente.
- **Faz sem orientação**. O estudante faz sem orientação quando realiza com a qualidade esperada, sem necessidade de apoio, uma determinada atividade. Nesse nível o estudante é capaz de dialogar com o docente sobre a atividade, sem depender dele para que possa ser realizada.

As condições mínimas não se confundem com as condições "desejadas", ou "ideais" de uma disciplina. De fato, espera-se que o egresso de cada disciplina esteja apto a realizar atividades além daquelas registradas nas condições mínimas e em níveis mais exigentes do que os estabelecidos. Contudo, ao realizar todas as atividades, nos níveis identificados pelas condições mínimas, o estudante é considerado aprovado na disciplina em questão.

Construção de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisitos:
NC	48	80	128	5.º	INF	Sim	Nenhum

Ementa

1. Visão geral (4h): construção (minimizar complexidade, antecipação de mudança, verificação, padrões), projeto de software (software design), qualidade de produto.
2. Planejamento (8h): linguagens de programação e processos de construção.
3. Gerência de construção (16h): controle de versão, inspeção e revisão de código.

4. Fundamentos de codificação (32h): estratégias recomendadas para criar código, variáveis, classes, interfaces, polimorfismo, rotinas, recursão, condições, laços, tratamento de exceção, reflexão, programação defensiva, padrão de codificação (leiaute e estilo), documentação, ferramentas de programação.
5. Projeto (design) detalhado e codificação (32h): noções de projeto detalhado, especificação de projeto, análise sintática (*parsing*), expressões regulares, parametrização (*generics*), *closure*, *logging*, configuração de software em tempo de execução. Internacionalização. Técnicas de construção baseadas em estado e tabelas.
6. Refatoração (8h).
7. Testes de unidade (16h).
8. Detecção e remoção de defeitos (debugging) (8h).
9. Integração (4h): integração contínua.

Condições mínimas (estar apto a):

- (Segue instruções). Explicar a construção de software, a relação dessa área de conhecimento com as demais da engenharia de software e a relação com o conceito de sistema.
- (Faz com orientação). Usar e configurar ferramentas para edição, documentação, compilação, depuração (*debugging*), *build*, teste, controle de versão, coleta de medidas (desempenho, consumo de memória, análise estática e cobertura), integração contínua, e aplicativos fundamentais de linha de comandos.
- (Faz com orientação). Criar código orientado a objetos em conformidade com o projeto (*design*) detalhado seguindo estratégias recomendadas.
- (Segue instruções). Usar processos para a construção de software (criação de código, controle de versão, inspeção e integração).
- (Segue instruções). Detalhar projeto (*design*) em conformidade com requisitos de software.
- (Segue instruções). Depurar (*debugging*) código.
- (Segue instruções). Criar testes de unidade básicos empregando estratégias recomendadas.
- (Segue instruções). Localizar e reutilizar código (bibliotecas e *frameworks*).

- (Segue instruções). Colaborar com a construção de código em equipe.
- (Segue instruções). Avaliar a qualidade interna de código e, quando apropriado, aplicar estratégias recomendadas de refatoração.

Modelagem de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisitos:
NC	16	48	64	5.º	INF	Sim	Nenhum

Ementa

1. Conceitos (4h): software, qualidade de software, requisitos de software, projeto de software.
2. Processo de projeto de software (4h).
3. Notações para registro de modelos orientados a objetos (16h).
4. Modelagem de software orientado a objetos (40h).

Condições mínimas (estar apto a)

- (Faz com orientação). Explicar o que é software, qualidade de software, requisitos de software e a relação entre eles.
- (Faz sem orientação). Empregar notações para registro de projeto de software orientado a objetos.
- (Segue instruções). Criar modelos de projeto de software orientado a objetos.

Processos de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	5.º	INF	Sim	Nenhum

Ementa

1. Visão geral (8h): processo, método e modelo.
2. Processos específicos de software: processos de implementação (12h), processos de suporte (12h) e processos de reutilização (2h).
3. Qualidade de Processo: modelos de qualidade de processos (12h), avaliação da qualidade do processo (12h) e abordagens de implementação de processo (6h).

Condições mínimas (estar apto a)

- (Segue instruções). Explicar processos de software, a relação dessa área de conhecimento com as demais da engenharia de software e a relação com o conceito de sistema.
- (Faz sem orientação). Correlacionar os conceitos de processo de software, método de software, modelo de software, modelo de processo e modelo de qualidade de processo.
- (Faz sem orientação). Explicar quais são os processos específicos de software: objetivo de cada processo e principais resultados.
- (Segue instruções). Explicar o uso de modelos de qualidade de processo: implementação e avaliação.
- (Segue instruções). Obter resultados de processo usando métodos e modelos.

Engenharia de Sistemas

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	64	0	64	5.º	INF	Sim	Nenhum

Ementa

1. Fundamentos de engenharia (16h): métodos empíricos e técnicas experimentais; análise estatística; medição; design; modelagem, prototipação e simulação; normas e padrões; análise de causa raiz.
2. Fundamentos de sistema (16h): taxonomias de sistema; sistemas de engenharia (sistemas produzidos por engenharia - *engineered systems*); sistemas intensivos em software (*software-intensive systems*); sistemas de sistemas; complexidade de sistemas; propriedades emergentes; princípios do pensamento sistêmico; representação de sistemas por modelos.
3. Engenharia de Sistemas (32h): abordagens e metodologias; partes interessadas (*stakeholders*) e suas necessidades; ciclo de vida de sistemas de engenharia (concepção, conceitos operacionais, design, validação de design, construção, validação de construção, implantação, sustentação e descontinuação); processos do ciclo de vida de sistemas (negociação, preparação para projeto organizacional, gestão técnica, processos técnicos); qualidade de processo.

Condições mínimas (estar apto a)

- (Segue instruções). Modelar o ciclo de vida de um sistema intensivo em software, considerando as implicações e perspectivas dos processos do ciclo de vida de sistemas.
- (Faz com orientação). Definir o conceito do sistema proposto (propósito desejado, contexto operacional, partes interessadas e conceito de uso do sistema).
- (Segue instruções). Selecionar um modelo de ciclo de vida de software adequado ao modelo de ciclo de vida de engenharia de sistemas e integrar os dois modelos.
- (Faz com orientação). Desenvolver conceitos operacionais do sistema (ambientes operacionais, características priorizadas, atributos de qualidade, cenários operacionais, suposições, dependências, limitações e exclusões).

Design de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	48	80	128	6.º	INF	Sim	Nenhum

Ementa:

1. Conceitos (8h): *design*, software, qualidade de software, requisitos de software, arquitetura de software e projeto detalhado.
2. Contexto de projeto de software (4h).
3. Processos e artefatos de projeto de software (8h).
4. Notações de projeto de software (8h).
5. Princípios e fundamentos de projeto de software (16h): abstração, acoplamento, coesão, decomposição, encapsulamento, separar interface e implementação, suficiência, completitude, simplicidade e *separation of concerns*.
6. Aspectos de projeto de software (16h): concorrência, controle e tratamento de eventos, tratamento de exceção, persistência, distribuição, interação (apresentação).
7. Qualidade de projeto de software (8h): técnicas de avaliação, medidas, ferramentas.
8. Métodos e estratégias de projeto (4h).
9. Método para projeto orientado a objetos (4h).

10. Padrões de projeto orientado a objetos (16h).
11. Prática de projeto de software orientado a objetos (36h).

Condições mínimas (estar apto a):

- (Faz sem orientação). Explicar o que é software, qualidade de software, requisitos de software, arquitetura de software, projeto detalhado, design, e a relação entre eles.
- (Faz com orientação). Aplicar princípios de projeto de software.
- (Faz com orientação). Empregar aspectos de projeto de software.
- (Faz com orientação). Fazer uso de medidas de qualidade de projeto de software.
- (Segue instruções). Criar projeto de software a partir de um problema.

Domínios de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	16	48	64	6.º	INF	Sim	Nenhum

Ementa

1. Visão ampla de domínios de desenvolvimento (4h): sistemas, componentes e a relação entre eles.
2. Especificidades de alguns domínios de desenvolvimento (8h).
3. Especificidades de um domínio (12h): requisitos, projeto, construção.
4. Desenvolvimento de software em um domínio (ênfase na construção) (40h).

Condições mínimas (estar apto a):

- (Faz com orientação). Explicar a construção de software, a relação dessa área de conhecimento com as demais da engenharia de software e a relação com o conceito de sistema.
- (Faz com orientação). Usar e configurar ferramentas para edição, documentação, compilação, depuração (debugging), build, teste, controle de versão, coleta de medidas (desempenho, consumo de memória, análise estática e cobertura), integração contínua, e aplicativos fundamentais de linha de comandos.
- (Faz com orientação). Criar código orientado a objetos em conformidade com o projeto (design) detalhado seguindo estratégias recomendadas.

- (Segue instruções). Usar processos para a construção de software (criação de código, controle de versão, inspeção e integração).
- (Segue instruções). Detalhar projeto (design) em conformidade com requisitos de software.
- (Segue instruções). Colaborar com a construção de código em equipe.
- (Segue instruções). Explicar domínios de desenvolvimento (categorias) e as especificidades correspondentes.
- (Segue instruções). Explicar um domínio de software e as estratégias de desenvolvimento correspondentes.

Processos de Qualidade de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	48	16	64	6.º	INF	Sim	Nenhum

Ementa

1. Visão Geral de Qualidade de Software (4h): custos e impactos da qualidade de software, questões éticas e culturais da qualidade de software.
2. Processo de garantia da qualidade de software (10h): garantia do produto e garantia do processo.
3. Processo de Verificação e Validação de Software (36h): requisito, projeto (design), código, integração, documentação.
4. Processos de Revisão e Auditoria de Software (14h): revisões gerenciais e revisões técnicas.

Condições mínimas (estar apto a)

- (Faz sem orientação). Explicar processos de qualidade de software, a relação dessa área de conhecimento com as demais da engenharia de software e a relação com o conceito de sistema.
- (Faz com orientação). Explicar a diferença e a relação entre a garantia do produto e a garantia do processo.
- (Segue instruções). Usar um processo da garantia da qualidade de software
- (Segue instruções). Usar um processo para a verificação e validação de software.

- (Segue instruções). Usar um processo para a revisão e auditoria de software.
- (Faz sem orientação). Colaborar em uma equipe na execução de um processo de qualidade.
- (Segue instruções). Documentar resultados dos processos de qualidade de software.
- (Segue instruções). Identificar e utilizar ferramentas de apoio ao controle de qualidade de artefatos de software.

Gerência de Projeto de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	6.º	INF	Sim	Nenhum

Ementa

1. Projeto e gerência de projeto (4h): ciclo de vida de produto, serviço e projeto; partes interessadas (stakeholders); correlação entre projetos, operações e programas; gerência de portfólio de projetos; escritório de projetos (PMO).
2. Tipos de projeto de software (4h): aquisição, desenvolvimento, refatoração, descontinuação.
3. Tipos de ciclo de vida de projeto de software (4h): preditivo, iterativo, adaptativo; processos empíricos e processos definidos e sua relação com métodos ágeis e métodos planejados para gerência de projeto de software.
4. Modelos de qualidade de processo de gerência de projeto de software (4h).
5. Áreas de conhecimento da gerência de projetos aplicadas a projetos de software (48h): escopo, tempo, custo, qualidade, recursos humanos, comunicações, riscos, aquisições, integração e partes interessadas.

Condições mínimas (estar apto a)

- (Segue instruções). Usar normas, modelos e ferramentas de gerência de projeto para planejamento e acompanhamento de projeto de software.
- (Faz com orientação). Selecionar e implementar o tipo de modelo de processo (dirigido por plano, incremental ou adaptativo) de acordo com as características do contexto do projeto de software.

- (Segue instruções). Selecionar e implementar o tipo de ciclo de vida de software (ou paradigma de engenharia de software - cascata, espiral, iterativo, baseado em modelo de maturidade, etc.) de acordo com as características do contexto do projeto de software.

Software Concorrente e Distribuído

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	7.º	INF	Sim	Nenhum

Ementa

1. Qualidade de produto (4h): escalabilidade, tolerância a falhas, disponibilidade, desempenho.
2. Conceitos (4h): middleware, distribuição, paralelismo, concorrência, RESTful, web, internet das coisas, contêineres, troca de mensagens, micro-serviços, serviços web.
3. Métodos para construção de software distribuído e para construção de software concorrente (8h).
4. Concorrência (16h): paralelismo, aplicações IO-intensive e CPU-intensive, thread safe, memória compartilhada, Software Transactional Memory, atores, abordagens para explorar concorrência.
5. Distribuição (24h): projeto, implementação e uso de interface de programação, RESTful, serviços web e micro-serviços (microservices).
6. Troca de mensagens (*messaging*) (8h).

Condições mínimas (estar apto a)

- (Faz com orientação). Explicar a construção de software, concorrência/distribuição e a relação desses com a engenharia de software e de sistema.
- (Faz com orientação). Usar e configurar ferramentas para edição, documentação, compilação, depuração (debugging), build, teste, controle de versão, coleta de medidas (desempenho, consumo de memória, análise estática e cobertura), integração contínua, e aplicativos fundamentais de linha de comandos.
- (Faz com orientação). Criar código orientado a objetos em conformidade com o projeto (design) detalhado seguindo estratégias recomendadas.

- (Segue instruções). Usar processos para a construção de software (criação de código, controle de versão, inspeção e integração).
- (Segue instruções). Detalhar projeto (design) em conformidade com requisitos de software.
- (Segue instruções). Localizar e reutilizar código (bibliotecas e frameworks).
- (Segue instruções). Colaborar com a construção de código em equipe.
- (Segue instruções). Avaliar a qualidade interna de código e, quando apropriado, aplicar estratégias recomendadas de refatoração.
- (Segue instruções). Explicar conceitos de concorrência e distribuição.
- (Segue instruções). Criar código que faz uso de concorrência, distribuição e ferramentas correlatas.

Experiência do Usuário de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	7.º	INF	Sim	Nenhum

Ementa

1. Qualidade em uso (4h): usabilidade, acessibilidade.
2. Design centrado no usuário (8h): diretrizes, pesquisa, personas, jornadas de usuário, modelagem (especificação) e ferramentas.
3. Projeto de interação (16h): diretrizes, criação, modelagem (especificação) e ferramentas.
4. Design de software e interfaces gráficas (8): padrões arquiteturais para design da camada de apresentação (MVC, MVP, MVVM e outros).
5. Implementação de interfaces (20h).
6. Modelos de validação de experiência (8h): modelos, métodos e técnicas para validação de experiência do usuário.

Condições mínimas (estar apto a)

- (Faz com orientação). Explicar a construção de software, experiência do usuário e a relação com a Engenharia de Software e a noção se sistema.
- (Faz com orientação). Usar e configurar ferramentas para edição, documentação, compilação, depuração (debugging), build, teste, controle de versão, coleta de medidas (desempenho, consumo de

memória, análise estática e cobertura), integração contínua, e aplicativos fundamentais de linha de comandos.

- (Faz com orientação). Criar código orientado a objetos em conformidade com o projeto (design) detalhado seguindo estratégias recomendadas.
- (Segue instruções). Usar processos para a construção de software (criação de código, controle de versão, inspeção e integração).
- (Segue instruções). Detalhar projeto (design) em conformidade com requisitos de software.
- (Segue instruções). Colaborar com a construção de código em equipe.
- (Segue instruções). Auxiliar na criação de requisitos de usabilidade.
- (Segue instruções). Criar e registrar projeto de interação que atende requisitos de usabilidade.
- (Faz com orientação). Usar ferramentas e bibliotecas para o desenvolvimento de código de interfaces.
- (Faz com orientação). Criar código em conformidade com o projeto de interação correspondente.

Arquitetura de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	7.º	INF	Sim	Nenhum

Ementa

1. Conceitos (4h): software, qualidade de software, requisitos de software, arquitetura de software e projeto detalhado.
2. Requisitos funcionais e requisitos de qualidade (8h): conceituação e identificação de requisitos relevantes para a arquitetura.
3. Documentação de arquitetura de software (8h).
4. Fundamentos de arquitetura de software (4h): importância, contexto, estilos.
5. Método de desenvolvimento de arquitetura de software (8h).
6. Técnicas para satisfazer requisitos de qualidade (16h).
7. Criação de arquiteturas de software (16h).

Condições mínimas (estar apto a)

- (Faz sem orientação). Explicar o que é software, qualidade de software, requisitos de software, arquitetura de software, projeto detalhado, e a relação entre eles.
- (Faz com orientação). Compreender a documentação de arquitetura de software.
- (Segue instruções). Documentar arquitetura de software.
- (Segue instruções). Criar arquitetura de software.

Requisitos de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	7.º	INF	Sim	Nenhum

Ementa

1. Processo de requisitos (16h): definições básicas, eliciação de requisitos, análise de requisitos, especificação de requisitos, verificação e validação de requisitos.
2. Modelos e métodos da engenharia de requisitos (32).
3. Gerência de projetos de engenharia de requisitos (8).
4. Processo de projeto arquitetural (8h): relação com requisitos.

Condições mínimas (estar apto a)

- (Faz sem orientação). Explicar requisitos de software, a relação dessa área de conhecimento com as demais da engenharia de software, arquitetura de software em especial, e a relação com o conceito de sistema.
- (Segue instruções). Usar um processo de requisitos de software.
- (Segue instruções). Eliciar de requisitos de software.
- (Segue instruções). Construir modelos conceituais de software.
- (Segue instruções). Especificar requisitos de software.
- (Faz com orientação). Verificar e validar requisitos de software.
- (Segue instruções). Identificar e utilizar ferramentas de requisitos de software.
- (Faz sem orientação). Colaborar em uma equipe na execução de um processo de requisitos.

Governança e Gestão de Serviços de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	48	16	64	7.º	INF	Sim	Nenhum

Ementa

1. Governança (8h): governança corporativa; governança de TI (GTI); governança de Software (GS); princípios da GS; conformidade (*compliance*); alinhamento dos serviços de software ao negócio da organização; normas, *frameworks*, padrões, modelos de qualidade e de maturidade de GS.
2. Abordagens para GS (8h): aplicação de abordagens de GS em um contexto organizacional. Objetivos de controle para GS; indicadores e avaliação de desempenho de software; direcionamento, avaliação e controle do uso de serviços de software para apoiar os objetivos de uma organização.
3. Planejamento e controle estratégico de software (8h): alinhamento entre objetivos organizacionais e objetivos relacionados a software; avaliação, direcionamento e monitoramento de processos de software; diagnóstico de maturidade de processos de software; alinhamento de políticas de software ao plano diretor de TI de uma organização. Gestão de riscos de software; riscos organizacionais; riscos relacionados à segurança física e lógica de software.
4. Governança de dados (8h): Segurança da informação em uma organização; Segurança de software.
5. Software como um serviço (SaaS) (8h): serviços de Software (SS); gestão de Serviços de Software (GSS); Estratégia organizacional para SS; Modelos, normas e padrões relacionados à GSS.
6. Ciclo de vida de SS (8h): planejamento; desenvolvimento; implantação e sustentação; Operação de SS e apoio (suporte) à operação de SS; Descontinuação de SS.
7. Manutenção de SS (8h): correção, adaptação e evolução de SS; gestão de incidentes e problemas em SS; controle de mudança e de configuração; controle de qualidade.
8. Gerência de processos de negócio (BPM – Business Process Management) aplicada à GSS (8h): representação de SS como processos

de negócio; notação para modelagem de processos de negócio (BPMN); análise, validação e evolução de processos de negócio e de SS.

Condições mínimas (estar apto a)

- (Segue instruções). Planejar a estratégia de software para uma organização, considerando a pouca disponibilidade de informações de longo prazo, a escassez de recursos para implementação dessa estratégia e a necessidade de alinhamento entre a GS e a governança corporativa.
- (Faz com orientação). Controlar o uso atual e futuro de software, otimizando o seu valor agregado para o negócio da organização.
- (Segue instruções). Planejar e implementar (projetar, desenvolver, implantar, monitorar, medir, controlar e sustentar) uma estratégia organizacional para GSS em conformidade com o plano estratégico da organização.
- (Faz com orientação). Modelar SS e processos de negócio, usando BPMN, em conformidade com os requisitos organizacionais dos processos e serviços.

Software para Sistemas Ubíquos

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	8.º	INF	Sim	Nenhum

Ementa

1. Sistemas de informação que fazem uso de dispositivos (ubíquos) (16h): smartphones, sensores, internet das coisas (IoT), *stream analytics* e aspectos de segurança (vulnerabilidades, criptografia, certificados digitais).
2. Definição de arquiteturas para soluções móveis (16): conectar serviços, possivelmente de grande volume, fluxo e em tempo real, com a necessidade de analisá-los.
3. Desenvolvimento de código para smartphone, sensor ou outro dispositivo capaz de alimentar/receber informações de sistema de informação (32h).

Condições mínimas (estar apto a):

- (Faz com orientação). Explicar a construção de software, dispositivos e a relação desses com a engenharia de software e de sistema.
- (Faz sem orientação). Explicar a construção de software e a relação dessa área de conhecimento com as demais da engenharia de software.
- (Faz com orientação). Usar e configurar ferramentas para edição, compilação, depuração (debugging), build, teste, controle de versão, coleta de medidas (desempenho, consumo de memória, análise estática e cobertura), integração contínua, e aplicativos de linha de comandos para busca e encadear comandos.
- (Faz com orientação). Criar código orientado a objetos em conformidade com o projeto (design) detalhado seguindo estratégias recomendadas.
- (Faz com orientação). Usar processos para a construção de software (criação de código, controle de versão, inspeção e integração).
- (Faz com orientação). Detalhar projeto (design) em conformidade com requisitos de software.
- (Faz com orientação). Detalhar projeto de interação (design) em conformidade com requisitos de software.
- (Faz com orientação). Localizar e reutilizar código (bibliotecas e frameworks).
- (Faz com orientação). Colaborar com a construção de código em equipe.
- (Faz com orientação). Explicar a construção de software, segurança e a relação desses com a engenharia de software e de sistema.
- (Segue instruções). Auxiliar na criação de modelo de ameaça.
- (Segue instruções). Criar código que faz uso de recursos de segurança (criptografia de dados, assinatura e verificação de assinatura digital).

Software para Persistência de Dados

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	8.º	INF	Sim	Nenhum

Ementa

1. Visão geral de persistência (12h): arquivos em formato texto/binário. Serialização de objetos. Documentos XML. JSON. Bancos de dados relacionais. Bancos NoSql.
2. Modelagem e implementação de bancos de dados (28h).
3. Mapeamento entre OO e relacional (8h).
4. Refatoração de bancos de dados (8h).
5. Integração de bancos de dados (8h).

Condições mínimas (estar apto a)

- (Faz com orientação). Explicar a construção de software, persistência e a relação desses com a engenharia de software e de sistema.
- (Segue instruções). Identificar e usar estratégia adequada de persistência de dados para um dado problema.
- (Faz com orientação). Usar ferramentas para definição e manipulação de bancos de dados.
- (Faz com orientação). Criar código que cria, busca, atualiza e remove dados em bancos de dados.
- (Segue instruções) Refatorar (*refactoring*) bases de dados.
- Faltou exigências de construção (anteriores).

Padrões de Arquitetura de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	8.º	INF	Sim	Nenhum

Ementa

1. Conceitos (12h): arquitetura de software e padrões de arquitetura (blackboard, microservices, invocação implícita, arquitetura orientada a serviços, multitier, arquitetura orienta a eventos, plugin, filtros e pipes, MVC.
2. Norma ISO/IEC/IEEE 42010:2011 (8h).
3. Análise (avaliação) de arquiteturas de software (12h).
4. Criação de arquiteturas de software (32h).

Condições mínimas (estar apto a)

- (Faz sem orientação). Compreender a documentação de arquitetura de software.

- (Faz com orientação). Explicar padrões de arquitetura e ilustrar o uso.
- (Faz com orientação). Documentar arquitetura de software.
- (Segue instruções). Realizar análise de arquitetura de software.
- (Segue instruções). Criar arquitetura de software que se beneficia de padrões de arquitetura.

Teste de Software

Núcleo	TEO	PRA	TOTAL	Período	Unidade	Obrigatória	Pré-requisito:
NC	32	32	64	8.º	INF	Sim	Nenhum

Ementa

1. Processo de construção (8h): definições básicas, atividades e documentação.
2. Processo de Teste de Software (56): definições básicas, técnicas de teste, teste baseado em intuição e experiência do engenheiro de software, atividades do processo, documentação e ferramentas.

Condições mínimas (estar apto a)

- (Faz sem orientação). Explicar o teste de software, a relação dessa área de conhecimento com as demais da engenharia de software e a relação com o conceito de sistema.
- (Segue instruções). Usar um processo de teste de software.
- (Segue instruções). Desenvolver planos de teste para o teste de unidade.
- (Faz com orientação). Propor casos de teste segundo algum critério da técnica de teste funcional para o teste de unidade.
- (Faz com orientação). Propor casos de teste segundo algum critério da técnica de teste estrutural para o teste de unidade.
- (Faz com orientação). Construir código para automação do teste de software.
- (Segue instruções). Aplicar teste exploratório no teste de sistema.
- (Faz sem orientação). Executar casos de teste.
- (Faz sem orientação). Citar e explicar métricas de teste de software.
- (Segue instruções). Realizar medições pertinentes a teste de software.
- (Faz com orientação). Documentar atividades do teste de software.
- (Segue instruções). Identificar e utilizar ferramentas de teste de software.

Metodologia e Experimentação em Engenharia de Software

Núcleo	Carga horária	Teórica	Prática	Período (fluxo)	Pré-requisito:
NC	64	48	80	8.º	Nenhum

Ementa

1. Metodologia Científica (16h): aplicação de metodologia científica em atividades de Engenharia de Software.
2. Fundamentos da Engenharia de Software Experimental (16h): contexto da experimentação na Engenharia de Software; principais tipos de experimentos aplicados à Engenharia de Software; estudos primários e secundários; revisão e mapeamento sistemático da literatura.
3. Realização de Experimentos em Engenharia de Software (32h).

Condições mínimas (estar apto a)

- (Faz com orientação). Aplicar princípios científicos da Engenharia de Software Experimental na realização de experimentos com software.
- (Segue instruções). Realizar estudo secundário sistemático em Engenharia de Software.

Mercado e Economia de Software

Núcleo	Carga horária	Teórica	Prática	Período (fluxo)	Pré-requisito:
NC	64	48	80	8.º	Nenhum

Ementa

1. Modelos de negócio para software (aluguel, serviço, código aberto).
2. Leis, normas, impostos e legislação brasileira para o mercado local e para a exportação de software.
3. Programas de incentivo à exportação e à produção de software.
4. Características e exigências do mercado interno e externo.
5. Identificação de oportunidades de inovação em software.
6. Planos de negócio de software para o mercado nacional e global.
7. Engenharia Econômica. Fornecimento, demanda e produção. Lucro produzido por capital (*interest*). Análise custo-benefício. Análise *breakeven*. Retorno de investimento. Avaliação de alternativas. Economia aplicada ao desenvolvimento de software.

Condições mínimas (estar apto a)

- (Segue instruções). Explicar a relação entre decisões técnicas e o alinhamento com objetivos de negócio no qual software está inserido.
- (Segue instruções). Explicar as principais leis e normas pertinentes à indústria de software no Brasil.
- (Segue instruções). Elaborar plano de negócio para software inovador.

Prática em Engenharia de Software

Núcleo	Carga horária	Teórica	Prática	Período (fluxo)	Pré-requisito:
NC	320	0	320	9.º	Nenhum

O egresso do BES passa pela vivência de dois tipos de experiências práticas diferentes em suas essências.

O primeiro grupo de experiências é composto pelas experiências promovidas nas atividades práticas realizadas de maneira contínua e integrada desde o início do curso, no contexto das diversas disciplinas teórico-práticas que compõem a grade curricular do BES.

O segundo tipo de experiências é proporcionado pela disciplina "Prática em Engenharia de Software", que envolve a participação integral do aluno em projetos reais de Engenharia de Software realizados em ambiente típico da indústria de software. É importante contrapor esse cenário de "projeto real", onde todas as competências são exigidas, com o cenário das atividades práticas realizadas dentro do contexto de uma disciplina, normalmente limitado pelo conteúdo da própria disciplina.

A Engenharia de Software compreende o uso de processos. Tais processos são claramente explicitados no projeto pedagógico do curso e podem ser classificados em dois grandes tipos: Processos Técnicos de Engenharia de Software e Processos de Gestão de Engenharia de Software. Além disso, o domínio de Tecnologias de Engenharia de Software, aplicadas tanto a processos técnicos quanto a processos de gestão de Engenharia de Software, é uma competência obrigatória para o profissional que trabalha com software. Assim, pode-se considerar que o Engenheiro de Software deve ser capaz de

realizar atividades que envolvem três tipos de competências: técnicas, gerenciais e tecnológicas.

A disciplina “Prática em Engenharia de Software” fornece o ambiente necessário para o amadurecimento das competências dos estudantes, pela aquisição de conhecimentos e desenvolvimento de habilidades e atitudes como profissional de Engenharia de Software. Esta prática acontece na fase final do currículo, na qual são reforçadas as habilidades, competências e conhecimentos adquiridos ao longo das disciplinas teórico-práticas do curso, em um ambiente que representa de forma realista os cenários que serão experimentados na vida profissional do Engenheiro de Software.

O INF possui uma Fábrica de Software (FS/INF) [CUSUMANO]. O termo “Fábrica” indica um comprometimento de longo prazo e de esforços integrados, acima do nível de projetos individuais, para aprimorar todas as operações de obtenção de software [AAEN]. NA FS/INF, os projetos têm como objetivo atender as necessidades de usuários e patrocinadores reais, ou seja, visam a geração de produtos e/ou a prestação de serviços em Engenharia de Software para a sociedade. Dessa forma, os projetos precisam atender, por exemplo, requisitos de qualidade, de escopo e restrições de custo e prazo definidos pelas partes interessadas do projeto em questão. Além disso, a FS/INF também é responsável por garantir os requisitos definidos pelas normas técnicas aplicáveis de Engenharia de Software.

Todas as atividades da Prática em Engenharia de Software previstas no BES são realizadas na FS/INF, sob a supervisão direta de docentes do Instituto de Informática. Portanto, a cada semestre letivo, a FS/INF recebe docentes e estudantes que atuarão nos seus projetos. Cabe ressaltar que, embora os projetos e as atividades da FS/INF não estejam limitados pelo calendário acadêmico, existe um compromisso entre a FS/INF e o curso de BES assegurando que o processo de Gestão do Portfólio da Fábrica selecionará os projetos e as atividades que apresentem condições adequadas ao exercício das práticas de Engenharia de Software pelos estudantes como descritas nesse PPC. Desta forma, o estudante terá a oportunidade concreta de integrar teoria e prática, envolvendo-se em situações-problema geradas pela experiência de campo e que exigem atividades de pesquisa, consultorias, debates e adoção de

novas condutas. É também uma oportunidade para que o estudante seja avaliado quanto à sua atitude ética e profissional, quanto ao respeito às normas institucionais da FS/INF e quanto à sua relação com os demais envolvidos no projeto, incluindo usuários e patrocinadores.

A “Prática em Engenharia de Software” é realizada pelo estudante preferencialmente após ter obtido aprovação nas demais disciplinas do BES. A disciplina é integralizada com 320 horas de atividades relacionadas a projetos da FS/INF, é ofertada a cada semestre letivo e os docentes atuam como preceptores. Neste sentido, o docente é o profissional responsável pela integração teoria-prática ao longo do projeto, ensinando, supervisionando, orientando e conduzindo o aluno na prática efetiva de sua futura profissão.

Os estudantes são divididos em três grupos para melhor acompanhamento das atividades. Cada grupo, em um dado instante, realiza um tipo de atividade típica do exercício profissional: processos técnicos, processos de gestão ou domínio de tecnologias. Ao final da disciplina, é garantido que todos os grupos vivenciaram atividades dos três tipos. A avaliação do estudante é formativa, enfatizando o seu acompanhamento durante todo o período letivo, com o intuito de não apenas verificar se o estudante está alcançando os objetivos propostos, mas também informando seus erros e acertos, além de promover o estímulo necessário para continuar os estudos e o seu desenvolvimento.

Concluindo, a disciplina “Prática em Engenharia de Software” favorece um ensino baseado na prática, com foco centrado na ética e na postura do profissional de Engenharia de Software perante os desafios que ocorrem quando se trabalha com software, que é um dos artefatos mais complexos que a humanidade produz. Quando o estudante atua em atividades práticas relevantes para a sua futura vida profissional, enfrentando os desafios reais de sua profissão ainda durante sua formação, sua compreensão se torna cada vez mais crítica e comprometida com a sociedade na qual se insere. Portanto, essa disciplina estimula e valoriza as dimensões ética e humanística na formação do Engenheiro de Software, desenvolvendo atitudes e valores orientados para a cidadania e para o desenvolvimento da sociedade.

Ementa

1. Aplicação do corpo de conhecimento da Engenharia de Software no contexto de projetos realizados em uma Fábrica de Software (320h): emprego de processos de Engenharia de Software em abrangência e profundidade; seleção e utilização de normas, métodos, técnicas e ferramentas de Engenharia de Software para atingir objetivos estabelecidos no projeto; integração e consolidação de conhecimentos e habilidades esperadas do profissional de Engenharia de Software; exercício de práticas e atitudes profissionais embasadas no código de ética e na postura profissional da Engenharia de Software; prática em processos técnicos de Engenharia de Software; prática em Gestão de Engenharia de Software; prática em Tecnologias de Engenharia de Software.

Condições mínimas (estar apto a)

- (Faz sem orientação). Realizar pacotes de trabalho, desenvolvendo produtos ou serviços dentro de um projeto na Fábrica de Software, aplicando as disciplinas que formam o corpo de conhecimento da Engenharia de Software.
- (Faz sem orientação). Planejar, realizar e modificar de forma apropriada produtos e serviços pertinentes a processos organizacionais da Engenharia de Software, dentro do contexto da Fábrica de Software.
- (Faz sem orientação). Atuar individualmente e nas equipes de trabalho segundo o código de ética e a postura profissional da Engenharia de Software.

Referências

- AAEN, I.; BØTTCHER, P.; MATHIASSEN, L. "The Software Factory: Contributions and Illusions", Proceedings of the Twentieth Information Systems Research Seminar. Scandinavia, Oslo, 1997.

- ACM/IEEE. Código de Ética e Prática Profissional da Engenharia de Software. Disponível em http://www.acm.org/about/se-code. Consultado em 20/07/2016.

- APOEMA. Apoema: Tecnologia e Inovação. Órgão complementar de pesquisa e desenvolvimento do Instituto de Informática. Disponível em http://apoema.inf.ufg.br.

- [BES1] *Bacharelado em Engenharia de Software na Universidade Federal de Goiás,* Fórum de Educação em Engenharia de Software (Simpósio Brasileiro de Engenharia de Software), número 43/08, páginas 16-21, Campinas/SP, 17/10/2008, ISSN 0103-9741.

- [BES2] *Engenharia de Software: Graduação (bacharelado) em Engenharia de Software,* Engenharia de Software Magazine, Ano I, 10a. edição, páginas 56-60, fevereiro de 2009, ISSN 1983127-7.

- CEPEC. RESOLUÇÃO CEPEC 1122/2012, 9 de novembro de 2012. Universidade Federal de Goiás (UFG), 2012.

- CEPEC. RESOLUÇÃO CEPEC 1286/2014, 6 de junho de 2014. Universidade Federal de Goiás (UFG), 2014.

- CNE. Conselho Nacional de Educação/Câmara de Educação Superior. Parecer CNE/CES 136/2012, 2012.

- COMTEC. Comunidade Tecnológica de Goiás (COMTEC). Disponível em: http://www.comtecgo.com.br/.

- CONSUNI. RESOLUÇÃO CONSUNI 10/2008, 27/06/2008. Universidade Federal de Goiás (UFG), 2008.

- CONSUNI. RESOLUÇÃO CONSUNI 02/2014, 24 de janeiro de 2014. Universidade Federal de Goiás (UFG), 2014.

- CUSUMANO, M. A. "Factory Concepts and Practices in Software Development". Annals of the History of Computing, 13(1): 3–32, 1991.
- IBGE. Portal http://www.ibge.gov.br/ consultado em 18/05/2014.
- [MEC 2012] Diretrizes Curriculares Nacionais para os cursos de graduação em Computação. Parecer CNE/CES 136/2012. Disponível em http://goo.gl/ks2R2O.
- OECD. Education at a Glance 2012: OECD indicators, OECD Publishing, 2012, ISBN 978-92-64-17929-5.
- PLS. Plano de Logística Sustentável da Universidade Federal de Goiás. Disponível em https://sustentabilidade.prodirh.ufg.br/.
- [SE 2014] Software Engineering 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. Disponível em http://www.acm.org/education/se2014.pdf.
- [SWEBOK 2014] B. Pierre e Fairley, R. E.. *Guide to the Software Engineering Body of Knowledge*, V3.0, IEEE Computer Society, 2014. Disponível em http://swebok.org.
- [SWECOM 2014] Software Engineering Competency Model (SWECOM), IEEE Computer Society, 2014. ISBN 978-0-7695-5373-3. Disponível em https://www.computer.org/web/peb/swecom.

Apêndice A

É prevista a migração de estudantes veteranos para o presente currículo (PPC). Nesses casos, a equivalência de disciplinas é realizada conforme as tabelas de equivalência seguintes.

Houve esforço no sentido de maximizar os aproveitamentos e, dessa forma, reduzir esforço necessário para integralização dos estudantes sem, contudo, prejudicar a formação desejada.

A primeira tabela define como disciplinas do PPC de 2009-1 podem ser aproveitadas no PPC proposto no presente documento (2017-1). Ou seja, apresenta como disciplinas já cursadas por veteranos poderão ser aproveitadas quando houver migração para o novo currículo.

A segunda tabela mostra como disciplinas do PPC de 2017-1 podem ser aproveitadas em termos de disciplinas do PPC de 2009-1.

Tabelas de equivalência

DISCIPLINA DO PPC DE 2009-1	CH	NATUREZA	APROVEITA COMO – NO PPC DE 2017-1	CH	NATUREZA
Introdução à Engenharia de Software	64	NE – OBR	Engenharia de Software	64	NE – OBR
Ética, Normas e Postura Profissional	64	NC – OBR	Computação e Sociedade	32	NC – OBR
Matemática Discreta	64	NC – OBR	Fundamentos de Matemática para Computação	64	NC – OBR
Lógica	64	NC – OBR	Lógica Matemática	64	NC – OBR
Introdução à Programação	64	NC – OBR	Introdução à Programação	128	NC – OBR
Método de Desenvolvimento de Software	64	NC – OBR	Programação Orientada a Objetos	64	NC – OBR
Construção de Software	64	NC – OBR	Domínios de Software	64	NC – OBR
Arquitetura de Computadores	64	NC – OBR	Arquitetura de Computadores	64	NC – OBR
Algoritmos: Fundamentos e Estruturas de Dados	64	NC – OBR	Algoritmos e Estruturas de Dados 1	64	NC – OBR
Banco de Dados	64	NC – OBR	Banco de Dados	64	NE – OBR
Engenharia de Software	64	NE – OBR	Engenharia de Software	64	NE – OBR
Sistema Operacional	64	NE – OBR	Sistemas Operacionais	64	NC – OBR
Redes e Sistemas Distribuídos	64	NE – OBR	Redes de Computadores	64	NC – OBR

DISCIPLINA DO PPC DE 2009-1	CH	NATUREZA	APROVEITA COMO – NO PPC DE 2017-1	CH	NATUREZA
Algoritmos: Ordenação e Busca	64	NC – OBR	Algoritmos e Estruturas de Dados 2	64	NC – OBR
Linguagens de Programação	64	NE – OBR	Linguagens e Paradigmas de Programação	64	NE – OPT
Requisitos de Software	64	NE – OBR	Requisitos de Software	64	NC – OBR
Projeto Detalhado de Software	64	NE – OBR	Projeto de Software	64	NE – OPT
Processo de Software	64	NE – OBR	Processos de Software	64	NE – OBR
Algoritmos em Grafos	64	NC – OBR	Algoritmos e Estruturas de Dados 2	64	NC – OBR
Interação Homem-Computador	64	NE – OBR	Interação Humano-Computador	64	NC – OBR
Segurança	64	NE – OBR	Optativa 1, Optativa 2 ou Optativa 3	64	NE – OBR
Qualidade de Software	64	NE – OBR	Processos de Qualidade de Software	64	NE – OPT
Arquitetura de Software	64	NE – OBR	Arquitetura de Software	64	NC – OBR
Verificação e Validação	64	NE – OBR	Testes de Software	64	NC – OBR
Gerência de Configuração de Software	64	NC – OBR	Optativa 1, Optativa 2 ou Optativa 3	64	NC – OBR
Manutenção de Software	64	NE – OBR	Optativa 1, Optativa 2 ou Optativa 3	64	NE – OPT
Gerência de Projeto de Software	64	NE – OBR	Gerência de Projeto de Software	64	NC – OBR
Métodos e Ferramentas da Engenharia de Software	64	NC – OBR	Optativa 1, Optativa 2 ou Optativa 3	64	NC – OBR
Experimentação em Engenharia de Software	64	NC – OBR	Metodologia e Experimentação em Engenharia de Software	64	NE – OBR
Leitura de Software	64	NE – OBR	Optativa 1, Optativa 2 ou Optativa 3	64	NC – OBR

DISCIPLINA DO PPC DE 2009-1	CH	NATUREZA	APROVEITA COMO – NO PPC DE 2017-1	CH	NATUREZA
Software Concorrente	64	NC – OBR	Software Concorrente e Distribuído	64	NC – OBR
Engenharia Econômica para Software	64	NC – OBR	Mercado e Economia de Software	64	NE – OBR
Software para Web	64	NE – OBR	Experiência do Usuário em Software	64	NE – OBR
Software para Dispositivos Móveis	64	NE – OBR	Software para Sistemas Ubíquos	64	NE – OBR
Software para Persistência	64	NC – OBR	Software para Persistência de Dados	64	NE – OBR
Integração de Aplicações **E**	64	NE – OBR	Design de Software	128	NE – OBR
Técnicas Avançadas de Construção de Software	64	NE – OBR			
Tópicos em Engenharia de Software	64	NE – OPT	Optativa 1, Optativa 2 ou Optativa 3	64	NC – OBR
LIBRAS	64	NE – OPT	Introdução à Língua Brasileira de Sinais	64	NE – OPT
Mercado de Software	64	NE – OPT	Mercado e Economia de Software	64	NE – OPT
Prática em Engenharia de Software	64	NE – OBR	Modelagem de Software	64	NE – OBR
Integração 1	64	NE – OBR	Padrões de Arquitetura de Software	64	NE – OBR
Integração 2	64	NE – OBR			

DISCIPLINA DO PPC DE 2017-1	CH	NATUREZA	APROVEITA COMO – NO PPC DE 2009-1	CH	NATUREZA
Computação e Sociedade	32	NC – OBR	Ética, Normas e Postura Profissional	64	NC - OBR
Introdução à Programação	128	NC – OBR	Introdução à Programação	64	NC - OBR
Cálculo 1A	64	NC – OBR	Optativa	64	NC – OBR
Fundamentos de Matemática para Computação	64	NC – OBR	Matemática Discreta	64	NC – OBR
Arquitetura de Computadores	64	NE – OPT	Arquitetura de Computadores	64	NC – OBR
Algoritmos e Estruturas de Dados 1	64	NE – OPT	Algoritmos: Fundamentos e Estruturas de Dados	64	NC – OBR
Probabilidade e Estatística A	64	NE – OBR	Optativa	64	NC – OBR
Álgebra Linear	64	NC – OBR	Optativa	64	NC – OBR
Lógica Matemática	64	NC – OBR	Lógica	64	NE – OBR
Programação Orientada a Objetos	64	NE – OBR	Método de Desenvolvimento de Software	64	NE – OBR
Algoritmos e Estruturas de Dados 2	64	NC – OBR	Algoritmos: Ordenação e Busca	64	NC – OBR
Optativa 1	64	NE – OPT	Optativa	64	NC – OBR
Linguagens e Paradigmas de Programação	64	NC – OBR	Linguagens de Programação	64	NE – OBR
Engenharia de Software	64	NE – OBR	Engenharia de Software	64	NC – OBR
Análise e Projeto de Algoritmos	64	NE – OBR	Algoritmos em Grafos	64	NC - OBR
Interação Humano-Computador	64	NE – OBR	Interação Homem-Computador	64	NE - OBR
Optativa 2	64	NE – OBR	Optativa	64	NC - OBR

DISCIPLINA DO PPC DE 2017-1	CH	NATUREZA	APROVEITA COMO – NO PPC DE 2009-1	CH	NATUREZA
Banco de Dados	64	NE – OBR	Banco de Dados	64	NC - OBR
Projeto de Software	64	NC – OBR	Projeto Detalhado de Software	64	NC - OBR
Construção de Software	128	NE – OBR	Introdução à Programação	64	NC – OBR
Modelagem de Software	64	NE – OPT	Optativa	64	NE – OBR
Processos de Software	64	NC – OBR	Processo de Software	64	NC – OBR
Engenharia de Sistemas	64	NE – OBR	Optativa	64	NE – OBR
Design de Software	128	NE – OBR	Projeto Detalhado de Software **E** Arquitetura de Software	64	NC – OBR
Domínios de Software	64	NE – OPT	Optativa	64	NE – OBR
Processos de Qualidade de Software	64	NE – OBR	Qualidade de Software	64	NC – OBR
Gerência de Projeto de Software	64	NE – OBR	Gerência de Projeto de Software	64	NE – OBR
Software Concorrente e Distribuído	64	NE – OBR	Software Concorrente	64	NC – OBR
Experiência do Usuário de Software	64	NE – OBR	Interação Homem-Computador	64	NC – OBR
Arquitetura de Software	64	NE – OBR	Arquitetura de Software	64	NC – OBR
Requisitos de Software	64	NC – OBR	Requisitos de Software	64	NE – OBR
Governança e Gestão de Serviços de Software	64	NE – OBR	Gerência de Configuração de Software	64	NE – OBR
Software para Sistemas Ubíquos	64	NE – OPT	Software para Dispositivos Móveis	64	NE – OPT

DISCIPLINA DO PPC DE 2017-1	CH	NATUREZA	APROVEITA COMO – NO PPC DE 2009-1	CH	NATUREZA
Software para Persistência de Dados	64	NE – OBR	Software para Persistência	64	NE - OBR
Padrões de Arquitetura de Software	64	NE – OBR	Arquitetura de Software **OU** Projeto Detalhado de Software	64	NE – OBR
Testes de Software	64	NE – OPT	Verificação e Validação	64	NC – OBR
Optativa 3	64	NE – OPT	Optativa	64	NC – OBR
Prática em Engenharia de Software	320	NE – OBR	-	64	NC – OBR
Sistemas Operacionais	64	NE – OPT	Sistema Operacional	64	NC - OBR
Pesquisa Operacional	64	NE – OPT	Optativa	64	NE – OBR
Linguagens Formais e Autômatos	64	NE – OBR	Optativa	64	NC – OBR
Redes de Computadores	64	NE – OBR	Redes e Sistemas Distribuídos	64	NC – OBR
Introdução à Língua Brasileira de Sinais	64	NE – OBR	LIBRAS	64	NC – OBR
Compiladores	64	NC – OBR	Optativa	64	NC – OBR
Sistemas Distribuídos	64	NE – OPT	Redes e Sistemas Distribuídos	64	NC – OBR
Mercado e Economia de Software	64	NC – OBR	Mercado de Software	64	NC - OBR
Metodologia e Experimentação em Engenharia de Software	64	NC – OBR	Experimentação em Engenharia de Software	64	NC - OBR
Design de Software	128	NE – OBR	Projeto Detalhado de Software **E** Arquitetura de Software	64	NC – OBR

DISCIPLINA DO PPC DE 2017-1	CH	NATUREZA	APROVEITA COMO – NO PPC DE 2009-1	CH	NATUREZA
Domínios de Software	64	NE – OPT	Optativa	64	NE – OBR
Processos de Qualidade de Software	64	NE – OBR	Qualidade de Software	64	NC – OBR
Gerência de Projeto de Software	64	NE – OBR	Gerência de Projeto de Software	64	NE – OBR
Software Concorrente e Distribuído	64	NE – OBR	Software Concorrente	64	NC – OBR
Experiência do Usuário de Software	64	NE – OBR	Interação Homem-Computador	64	NC – OBR
Arquitetura de Software	64	NE – OBR	Arquitetura de Software	64	NC – OBR
Requisitos de Software	64	NC – OBR	Requisitos de Software	64	NE – OBR
Governança e Gestão de Serviços de Software	64	NE – OBR	Gerência de Configuração de Software	64	NE – OBR
Software para Sistemas Ubíquos	64	NE – OPT	Software para Dispositivos Móveis	64	NE – OBR
Software para Persistência de Dados	64	NE – OBR	Software para Persistência	64	NE – OBR
Padrões de Arquitetura de Software	64	NE – OBR	Arquitetura de Software **OU** Projeto Detalhado de Software	64	NE – OBR
Testes de Software	64	NE – OPT	Verificação e Validação	64	NE – OBR
Optativa 3	64	NE – OPT	Optativa	64	NE – OBR
Prática em Engenharia de Software	320	NE – OBR	-	64	NE – OBR
Sistemas Operacionais	64	NE – OPT	Sistema Operacional	64	NC – OBR
Pesquisa Operacional	64	NE – OPT	Optativa	64	NE – OBR

DISCIPLINA DO PPC DE 2017-1	CH	NATUREZA	APROVEITA COMO – NO PPC DE 2009-1	CH	NATUREZA
Linguagens Formais e Autômatos	64	NE – OBR	Optativa	64	NC – OBR
Redes de Computadores	64	NE – OBR	Redes e Sistemas Distribuídos	64	NC – OBR
Introdução à Língua Brasileira de Sinais	64	NE – OBR	LIBRAS	64	NC – OBR
Compiladores	64	NE – OPT	Optativa	64	NC – OBR
Sistemas Distribuídos	64	NE – OPT	Redes e Sistemas Distribuídos	64	NC – OBR
Mercado e Economia de Software	64	NC – OBR	Mercado de Software	64	NC - OBR
Metodologia e Experimentação em Engenharia de Software	64	NC – OBR	Experimentação em Engenharia de Software	64	NC - OBR

Apêndice B

Reúne as bibliografias básicas e complementares das disciplinas do curso. As disciplinas estão agrupadas em disciplinas de formação básica (obrigatórias), disciplinas de formação básica (optativas) e disciplinas específicas de Engenharia de Software.

Disciplinas de formação básica (obrigatórias)

Computação e Sociedade

Bibliografia Básica

- FONSECA FILHO, C. História da computação: O Caminho do Pensamento e da Tecnologia. Porto Alegre: EDIPUCRS, 2007.
- MASIERO, P. Ética em Computação. Editora da USP, 2000.
- VELOSO, R. Tecnologias da Informação e Comunicação: Desafios e Perspectivas. São Paulo: Saraiva, 2011.

Bibliografia Complementar

- CHALITA, G. Os Dez Mandamentos da Ética. Editora Nova Fronteira, 2003.
- DRUMMOND, V. Internet, Privacidade e Dados Pessoais. Editora Lumen Juris, 2003.
- KACZMARCZYK, L. C. Computers and Society: Computing for Good. Chapman & Hall/CRC Textbooks in Computing. CRC Press, 2011.
- LUCCA, N.; FILHO, A. S. Direito & Internet. Editora Edipro, 2001.
- PAESANI, L. M. Direito de Informática. Editora Atlas, 2005.

Introdução à Programação

Bibliografia básica

- FOBERLONE, A. L. V.; EBERSPACHER, H. F. Lógica de Programação: A construção de algoritmos e estruturas de dados. 3.ª edição. São Paulo: Prentice Hall, 2005.
- ASCENCIO, A. F. G.; CAMPOS, E. A. V. Fundamentos da Programação de Computadores. 3.ª edição. Editora Pearson, 2010.
- SCHILDT, H. C Completo e Total. 3a Ed. São Paulo: Makron Books, 1996.

Bibliografia complementar

- FEOFILOFF, P. Algoritmos em Linguagem C. Editora Campus/Elsevier, 2009.
- FARRER, H. at al. Programação Estruturada de Computadores: Algoritmos Estruturados. 3.ª edição. Rio de Janeiro: LTC, 1989.
- SEDGEWICK, R. Algorithms in C. 3rd ed. Reading, Mss: Addison-Wesley, 1998. ISBN 0201314525.
- SALVETTI, D. D.; BARBOSA, L. M. Algoritmos, São Paulo: Makron Books, 1998.
- CORMEN, T. H et al., Algoritmos: Teoria e Prática. 2.ª edição. Rio de Janeiro: Editora Campus, 2002.

Cálculo 1A

Bibliografia Básica

- GUIDORIZZI, H. L. Um Curso de Cálculo, vol. 1. LTC, Rio de Janeiro, Brasil, 2006.
- LEITHOLD, L. O Cálculo com Geometria Analítica, 3.ª edição. vol. 1. Harbra, São Paulo, 1994.
- STEWART, J. Cálculo, 5.ª edição. vol. 1. Cengage Learning, São Paulo, 2006.

Bibliografia Complementar

- ÁVILA, G. S. S. Cálculo: Funções de Uma Variável, 7.ª edição. vol. 1. LTC: Rio de Janeiro, 1994.
- FLEMMING, D. M.; GONÇALVES, M. B. Cálculo A: Funções, limite, derivação e integração. Makron Books do Brasil, São Paulo, 2006.
- HOFFMANN, L. D.; BRADLEY, G. L. Cálculo: Um curso moderno com aplicações, 9.ª edição. LTC, Rio de Janeiro, 2008.
- REIS, G. L; SILVA, V. V. Geometria Analítica. LTC, São Paulo.
- SIMMONS, G. F. Cálculo com Geometria Analítica, vol. 1. McGraw-Hill do Brasil, São Paulo, Brasil, 1987.

Fundamentos de Matemática para Computação

Bibliografia Básica

- GERSTING, J. L. Fundamentos Matemáticos para a Ciência da Computação, 5.ª edição, Editora LTC, Rio de Janeiro, 2004.
- SCHEINERMAN, E. R. Matemática Discreta: Uma introdução, Cengage Learning, Thompson Pioneira, 2003
- GRIMALDI, R. P. Discrete and Combinatorial Mathematics: An Applied Introduction, 5th edition, Pearson - Addison-Wesley, Boston, 2003.

Bibliografia Complementar

- GONÇALVES, A. Introdução à Álgebra, 5.ª edição, CNPQ-IMPA, Rio de Janeiro, 2013.
- MATTSON, Jr.; H. F. Discrete Mathematics with applications, John Wiley & Sons, New York, 1993.
- ROSEN, K. H. Discrete Mathematics and Its Applications, 6.ª edição, Boston: McGraw-Hill, 2009.
- ROSS, K. A.; Wright, C. R. B. Discrete mathematics, Prentice Hall, 1998.
- SANTOS, J. P. O. Introdução à Teoria dos Números, Coleção Matemática Universitária, CNPQ-IMPA, Rio de Janeiro, 2003.

Arquitetura de Computadores

Bibliografia Básica

- BRYANT, R.; O'RALLARON, D. Computer Systems: A Programmer's Perspective, 2nd Edition, Addison Wesley, 2010.
- STALLINGS, W. Arquitetura e Organização de Computadores, 5a. Edição, Prentice-Hall, 2002.
- TANENBAUM, A. Organização Estruturada de Computadores, Editora LTC, 2006.

Bibliografia Complementar

- HENNESSY, J. L.; PATTERSON, D. A. Computer Architecture: A Quantitative Approach, 4th Edition, Morgan Kaufmann, 2007.
- MONTEIRO, M. A. Introdução à Organização de Computadores. 4ª. Edição. Editora LTC, Rio de Janeiro, 2001. ISBN: 8521612915.
- PATTERSON, D. A.; HENNESSY, J. L. Computer Organization and Design: The Hardware/Software Interface, 3rd edition, Morgan Kaufmann, 2007.
- STALLINGS, W. Computer Organization and Architecture: Designing for Performance. 10th Edition. Prentice-Hall, Inc., Upper Saddle River, NJ, USA. 2010.
- WEBER, R. F. Fundamentos de Arquiteturas de Computadores, 2.ª Edição, Editora Sagra-Luzzatto, 2001.

Algoritmos e Estruturas de Dados 1

Bibliografia Básica

- FEOFILOFF, P. Algoritmos em Linguagem C. Editora Campus/Elsevier, 2009.
- SZWARCFITER, J. L.; MARKENZON, L. Estruturas de Dados e seus Algoritmos. 2ª edição, LTC, 1994.
- TENENBAUM, A. M.; LANGSAM, Y.; AUGENSTEIN, M. Estruturas de Dados Usando C, São Paulo, Makron Books, 1995.

Bibliografia Complementar

- CORMEN, T. H. et al., Algoritmos: Teoria e Prática. 2.ª edição, Rio de Janeiro: Editora Campus, 2002.

- SALVETTI, D. D.; BARBOSA, L. M.; Algoritmos, Makron Books, São Paulo, 1998.
- SEDGEWICK, R. Algorithms in C++ (Parts 1-4), Addison-Wesley, 3rd Edition, 1998.
- ZIVIANI, N. Projeto de Algoritmos com implementação em Pascal e C. São Paulo: Editora Thomson, 3.ª edição, 2010.
- ZIVIANI, N. Projeto de Algoritmos com implementação em Java e C++. São Paulo: Editora Thomson, 2006.

Probabilidade e Estatística A

Bibliografia Básica

- MAGALHÃES, M. N. Noções de Probabilidade e Estatística. São Paulo: EDUSP, 7.ª ed., 2010.
- MEYER, P. L. Probabilidade: Aplicações à Estatística. Rio de Janeiro: LTC, 1969.
- WALPOLE, R. E.; MYERS, R. H.; MYERS, S. L.; YE, K. Probabilidade e Estatística para Engenharia e Ciências. São Paulo: Pearson, 8.ª ed., 2009.

Bibliografia Complementar

- BUSSAB, W. O.; MORETTIN, P. A. Estatística Básica. 6.ª ed. São Paulo: Saraiva, 2010.
- DANTAS, C. A. B. Probabilidade: um Curso Introdutório. 3.ª ed. São Paulo: EDUSP, 2008.
- MORETTIN, L. G. Estatística básica: probabilidade e inferência. São Paulo: Prentice Hall, 2010.
- ROSS, S. Probabilidade. Um Curso Moderno com Aplicações. 8.ª ed. Porto Alegre: Bookman, 2010.
- TRIOLA, M. F. Introdução à Estatística. 10. ed. Rio de Janeiro: LTC, 2008.

Álgebra Linear

Bibliografia Básica

- BOLDRINI, J. L.; COSTA, S. I. R; FIGUEIREDO, V. L; WETZLER, H. G. Álgebra Linear, 3.ª ed. Harbra, São Paulo, 2003.
- CALLIOLI, C. A.; DOMINGUES, H. H; COSTA, R. C. F. Álgebra Linear e Aplicações. Atual, Brasil, 1983.
- LIPSCHUTZ, S. Álgebra Linear, 2.ª ed. Makrom-Books, São Paulo, Brasil, 1974.

Bibliografia Complementar

- APOSTOL, T. Linear Algebra: A First Course with Applications to Differential Equations, 1st ed. Wiley-Interscience, 1997.
- HOFFMAN, K.; KUNZE, R. Álgebra Linear. Polígono, São Paulo, 1971.
- HOWARD, A.; RORRES, C. Álgebra Linear com Aplicações, 8.ª ed. Bookman, Porto Alegre, Brasil, 2001.
- LIMA, E. L. Álgebra Linear: Coleção Matemática Universitária. IMPA, Rio de Janeiro, Brasil, 2006.
- SHOKRANIAN, S. Introdução à Álgebra Linear e Aplicações, 1 ed. UNB, 2004.

Lógica Matemática

Bibliografia Básica

- SOUZA, J. N. Lógica para Ciência da Computação. Editora Campus, 3.ª Edição, 2015.
- SILVA, F. C.; FINGER, M.; MELO, A. C. V. Lógica para Computação. Thomson Learning, 2006.
- HUTH, M.; RYAN, M. Lógica em Ciência da Computação: modelagem e argumentação sobre sistemas. 2.ª edição. Editora LTC, 2008.

Bibliografia Complementar

- MORTARI, C. Introdução à Lógica. São Paulo: UNESP. 2001.
- MENDELSON, E. Introduction to Mathematical Logic, Academic Press, 2000.
- ENDERTON, H. A. Mathematical Introduction to Logic. Academic Press 2000.

- SMULLYAN, R. Lógica de Primeira Ordem. São Paulo: UNESP. 2009.
- CASANOVA, M. A.; GIORNO, F. A. C.; FURTADO, A. L. Programação em Lógica e a Linguagem PROLOG. Edgard Blucher, 1987.

Programação Orientada a Objetos

Bibliografia Básica

- BORATTI, I. C. Programação orientada a objetos em Java. 1.ª Edição. Visual Books, 2007.
- ECKEL, B. Thinking in Java. Prentice Hall, 3.ª Edição, 2002.
- DEITEL, P. J.; DEITEL, H. M. Java: como programar 6.ª ed. São Paulo: Prentice Hall, 2005. 1386 p.

Bibliografia Complementar

- BUDD, T. An Introduction to Object-Oriented Programming. Addison Wesley, 1996.
- GAMMA, E. et al. Design patterns: elements of reusable object-oriented software. Reading: Addison Wesley, 1995.
- HORSTMANN, C. S. Core Java – Volume II – Advanced Features, Prentice Hall, 8.ª Edição, 2008.
- SANTOS, R. Introdução à Programação Orientada a Objetos com Java. Campus, 2003.
- ZEIGLER, B. P. Objects and Systems: Principled Design with Implementations in C++ and Java. Springer-Verlag New York, Inc., New York, NY, USA. 1997

Algoritmos e Estruturas de Dados 2

Bibliografia Básica

- FEOFILOFF, P. Algoritmos em Linguagem C. Editora Campus/Elsevier, 2009.
- SZWARCFITER, J. L.; Markenzon, L. Estruturas de Dados e seus Algoritmos. 2ª edição, LTC, 1994.
- TENENBAUM, A. M.; LANGSAM, Y.; AUGENSTEIN, M. Estruturas de Dados Usando C, São Paulo, Makron Books, 1995.

Bibliografia Complementar

- CORMEN, T. H. et al., Algoritmos: Teoria e Prática. 2ª edição, Rio de Janeiro: Editora Campus, 2002.
- SALVETTI, D. D. e BARBOSA, L. M., Algoritmos, Makron Books, São Paulo, 1998.
- SEDGEWICK, R. Algorithms in C++ (Parts 1-4), 3rd edition, Addison-Wesley, 1998.
- ZIVIANI, N. Projeto de Algoritmos com implementação em Pascal e C. 3ª edição, São Paulo: Editora Thomson, 2010.
- ZIVIANI, N. Projeto de Algoritmos com implementação em Java e C++. São Paulo: Editora Thomson, 2006.

Linguagens e Paradigmas de Programação

Bibliografia Básica

- GHEZZI, C.; JAZYERI, M. Programming Language Concepts, 3rd Edition, Wiley; 3rd Edition; 1997.
- SCOTT, M. L. S. Programming Language Pragmatics, 3rd Edition, Morgan Kaufmann, 2009.
- SEBESTA, R. W. Concepts of Programming Languages; 10th Edition, Pearson, 2012.

Bibliografia Complementar

- FRIEDMAN, D. P.; WAND, M. Essentials of Programming Languages, 3rd Edition, The MIT Press, 2008.
- PRATT, T. W.; ZELKOWITZ, M. V. Programming Languages: Design and Implementation, 4th Edition, Prentice Hall, 2000.
- ROY, P. V.; HARIFI, S. Concepts, Techniques, and Models of Computer Programming, 1st Edition, The MIT Press, 2004.
- TURBAK, F.; GIFFORD, D. Design Concepts in Programming Languages, 1st Edition, The MIT Press, 2008.
- VAREJÃO, F. Linguagens de Programação, 1.ª Edição, Editora. Campus, 2004.

Engenharia de Software

Bibliografia Básica

- PRESSMAN, R.; MAXIM, B. Software Engineering: A Practitioner's Approach, 8th edition, McGraw-Hill, 2014.
- SOMMERVILLE, I. Software Engineering, 10th Edition, Pearson, 2016.
- WAZLAWICK, R. S. Engenharia de software: teoria e prática. Campus, 2013.

Bibliografia Complementar

- ISO/IEC/IEEE 24765. Systems and software engineering - Vocabulary, 2010. Portal de periódicos da CAPES (http://goo.gl/FrGm7c).
- MAGELA, R. Engenharia de software aplicada. Alta Books, 2006.
- MPS.BR. Melhoria de Processos do Software Brasileiro. Disponível em http://www.softex.br/mpsbr/, último acesso julho/2016.
- Software Engineering Body of Knowledge (SWEBOK), V3, IEEE, Disponível em https://www.computer.org/web/swebok.
- Software Engineering Competency Model (SWECOM), IEEE. Disponível em https://www.computer.org/web/peb/swecom.

Análise e Projeto de Algoritmos

Bibliografia Básica

- BRASSARD, G.; BRATLEY, P. Fundamentals of Algorithmics. Prentice-Hall, Inc., Upper Saddle River, NJ, 1996. ISBN: 0-13-335068-1.
- CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L.; STEIN, C. Algoritmos: Teoria e Prática, 3.ª edição, Campus, 2012.
- PAPADIMITRIOU, C. H.; VAZIRANI, U. V.; DASGUPTA, S. Algoritmos. 2009. McGraw-Hill Brasil. ISBN 9788577260324.

Bibliografia Complementar

- AHO, A. V.; HOPCROFT, J. E.; ULLMAN, J. D. The Design and Analysis of Computer Algorithms, Addison-Wesley Publishing Company, 1974. ISBN 0-201-00029-6.
- BAASE, S.; GELDER, A. V. Computer Algorithms: Introduction to Design and Analysis, 3rd Edition, Pearson, 1999.

- MAMBER, U. Introduction to Algorithms. Addison Wesley Publishing Company. 1989.
- SEDGEWICK, R.; WAYNE, K. Algorithms. 4th edition, Addison-Wesley Professional, 2011. ISBN: 978-0321573513
- SZWARCFITER, J. L.; MARKENZON, L. Estrutura de Dados e seus Algoritmos. 3.ª edição, LTC Editora, 2010. ISBN 978852161750.

Interação Humano-Computador

Bibliografia Básica

- DIX, A.; FINLAY, J. E.; ABOWD, G. D.; BEALE, R. Human-Computer Interaction (3rd Edition). Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2003.
- ROGERS, Y.; SHARP, H.; PREECE, J. Design de Interação - Além da Interação Homem-computador. 3a Ed., Porto Alegre: Bookman, 2013.
- TIDWELL, J. Designing Interfaces: Patterns for Effective Interaction Design. O'Reilly. Segunda edição, 2011.

Bibliografia Complementar

- BARBOSA, S. D. J.; SILVA, B. S. Interação Humano-Computador. Elsevier Editora. 1.ª edição. 2010.
- JACKO, Julie A. The human-computer interaction handbook: fundamentals, evolving technologies, and emerging applications. CRC Press, 2012.
- LAZAR, J.; FENG, J. H.; HOCHHEISER, H. Research Methods in Human-Computer Interaction. Wiley, 2009.
- MACKENZIE, I. S. Human-computer Interaction: An Empirical Research Perspective. Morgan Kaufmann, 201 3. ISBN: 978-0-1 2-405865-1
- SCOTT, B.; NEIL, T. Designing Web Interfaces: Principles and Patterns for Rich Interactions. O'Reilly Media; 1st edition. 2009.

Banco de Dados

Bibliografia Básica

- ELMASRI, R.; NAVATHE, S. B. Sistemas de Banco de Dados, 6.ª ed., Pearson - Addison Wesley, 2011.
- HEUSER, C. A. Projeto de Banco de Dados, 6.ª edição, Bookman, Porto Alegre, 2009.
- SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. Sistema de Banco de Dados, 5.ª edição, Editora Campus, Rio de Janeiro, 2006.

Bibliografia Complementar

- CONNOLLY, T. M.; BEGG, C. E.; STRACHAN, A. D. Database systems: a practical approach to design, implementation and management, 3rd. Edition, Addison Wesley, 2010.
- DATE, C. J. Introdução a Sistemas de Banco de Dados, tradução da 8.ª edição americana, Editora Campus, Rio de Janeiro, 2004.
- GARCIA-MOLINA, H.; ULLMAN, J. D.; WIDOM, J. D. Database Systems: The Complete Book, 2.ª edição, Prentice Hall, 2009
- RAMAKRISHNAN, R.; GEHRKE, J. Sistemas de Gerenciamento de Banco de Dados, tradução da 3.ª edição, São Paulo, McGraw-Hill, 2008.
- TEOREY, T.; LIGHTSTONE, S.; NADEAU, T. Projeto e Modelagem de Bancos de Dados, Editora Campus, Rio de Janeiro, 2007.

Projeto de Software

Bibliografia Básica

- BUDGEN, B. Software Design, 2nd edition, Addison-Wesley, 2003.
- BASS, L. et al. Software Architecture in Practice, 3rd edition, Addison-Wesley, 2012.
- LARMAN, C. Utilizando UML e padrões: uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento interativo. Bookman, 2008.

Bibliografia Complementar

- BOOCH, G. UML guia do usuário: o mais avançado tutorial sobre Unified Modeling Language (UML). Elsevier, 2006.

- FREEMAN, E. et al. Use a cabeça!: padrões de projetos, (design patterns), Alta Books, 2007.
- FOWLER, M. Patterns of enterprise application architecture. Addison-Wesley, 2003.
- GAMMA, E. et al. Design patterns elements of reusable object-oriented software. Reading: Addison Wesley, 1995.
- NYGARD, M. Release It!: Design and Deploy Production-Ready Software, Pragmatic Bookshelf, 2007.

Disciplinas optativas

Pesquisa Operacional

Bibliografia básica

- BREGALDA, P. F.; OLIVEIRA, A. A. F.; BORNSTEIN, C. T. Introdução à programação linear. 3. ed. Rio de Janeiro: Campus, 1983.
- GOLBARG, M.; LUNA, H. Otimização combinatória e programação linear. 2. ed. Campus, 2005.
- TAHA, H. Pesquisa operacional. 8. ed. Prentice Hall, 2008.

Bibliografia complementar

- BAZARAA M. S.; JARVIS, J. J.; SHERALI, H. D. Linear programming and network Flows. Wiley, 2009.
- HILLIER, F. S.; LIEBERMAN, G. J. Introdução à pesquisa operacional. 9.ª edição. Amgh Editora, 2013.
- PARLAR, M. Interactive operations research with Maple: methods and models. Birkhauser, 2000.
- SILVA, E. M. et al. Pesquisa Operacional: programação linear, simulação. 3.ª ed. Atlas, 1998.
- WINSTON, W. L. Operations research applications and algorithms. 3rd ed. Duxbury Press, 1997.

Linguagens Formais e Autômatos

Bibliografia básica

- SUDKAMP, T. A. Languages and machines. 3.ª edição. Addison Wesley, 2006.
- SIPSER, M. Introduction to the theory of computation. ITP, 1997. ISBN 053494728X.
- HOPCROFT, J. E.; MOTWANI, R.; ULLMAN, J. D. Introdução à teoria de autômatos: linguagens e computação. Campus, 2002.

Bibliografia complementar

- RAMOS, M. V. M.; NETO, J. J.; VEGA, I. S. Linguagens formais: teoria, modelagem e implementação. Bookman, 2009.
- CARROLL, J.; LONG, D. Theory of finite automata. New Jersey: Prentice-Hall International Editions, 1989.
- LEWIS, H. R.; PAPADIMITRIOU, C. H. Elementos de teoria da computação. 2.ª edição. Porto Alegre: Bookman, 2000.
- MENEZES, P. F. B. Linguagens formais e autônomos. 5.ª edição. Porto Alegre: Sagra Luzzatto, 2005.
- ROSEN, K. H. Matemática discreta e suas aplicações. 6.ª ed. Rio de Janeiro: McGraw Hill, 2009.

Sistemas Operacionais

Bibliografia Básica

- OLIVEIRA, R. S.; CARISSIMI, A. S.; TOSCANI, S. S. Sistemas Operacionais. 3.ª edição. Porto Alegre: Instituto de Informática da UFRGS: Editora Sagra Luzzatto, 2004.
- SILBERSCHATZ, A. Fundamentos de Sistemas Operacionais, 8.ª edição, LTC, 2011.
- TANEMBAUM, A. S. Sistemas Operacionais Modernos. 3.ª edição. São Paulo: Editora Prentice Hall, 2010. ISBN 9788576052371.

Bibliografia Complementar

- DEITEL, H. Sistemas Operacionais, Prentice Hall, 2005.
- MACHADO, F. B. Arquitetura de Sistemas Operacionais. 2.ª edição. LTC, 1997.

- NEMETH E.; SNYDER, G.; HEIN, T. R. Unix system administration handbook. Pearson Prentice Hall, 1997. ISBN 9788576051121.
- O'GORMAN, J. Operating Systems with Linux. Palgrave, 2001.
- SHAY, W. Sistemas Operacionais, Makron Books, 1996.

Compiladores

Bibliografia básica

- AHO, A. V; LAM, M. S.; SETHI, R.; ULLMAN, J. D. Compiladores: Princípios, técnicas e ferramentas. 2.ª ed. Pearson-Addison-Wesley, 2008.
- LOUDEN, K. C. Compiladores - Princípios e Práticas. Editora Thompson 2004.
- APPEL, A. W. Modern Compiler Implementation in C - Basic Techiniques. Cambridge University Press, 1997.

Bibliografia complementar

- APPEL, A. W. Modern Compiler Implementation in Java. 2nd edition. New York: Cambridge University Press, 2002.
- HOLMES, J. Modern Compiler Implementation in Java. Prentice Hall, 1995.
- MUCHNICK, S. S. Advanced Compiler Design and Implementation. Morgan Kaufmann, 1997.
- HANSON, D. R.; FRASER, C. W. A Retargetable C Compiler: Design and Implementation. Benjamin Cummings Pub., 1995.
- LEBLANC, R. J.; FISCHER, C. N. Crafting a Compiler with C. Benjamin/Cummings, 1991.

Redes de computadores

Bibliografia básica

- KUROSE, J.F.; ROSS, K. Redes de Computadores e a Internet, 6.ª edição. Pearson Education, 2013.
- LAUDON, K. C.; LAUDON, J. P. Sistemas de Informação Gerenciais. 9.ª edição. São Paulo: Pearson Education, 2013.

- TANENBAUM, A. S. Computer Networks, 4th edition, Prentice Hall, 2003.

Bibliografia complementar

- DANTAS, M. Redes de comunicação e computadores: abordagem quantitativa. Visual Books, 2009. ISBN 9788575022559.
- PETERSON, L. L.; DAVIE, B. S. Redes de Computadores: Uma Abordagem de Sistemas, 3.ª edição. Campus Elsevier, 2004.
- SOARES, L. F. G; SOUZA FILHO, G. L.; COLCHER, S. Redes de computadores: das LANS, MANS e WANS às Redes ATM. Editora Campus, 1995. ISBN 857001998X.
- STALLINGS, W. Data and Computer Communications, 8th edition, Pearson/Prentice Hall, 2007.
- TORRES, G. Redes de computadores: curso completo. Axcel Books, 2001.

Introdução à Língua Brasileira de Sinais

Bibliografia Básica

- FELIPE, T.; MONTEIRO, M. S. LIBRAS em contexto. Curso Básico. Brasília: Ministério da Educação e do Desporto/Secretaria de Educação Especial, 2001.
- PEREIRA, M. C. C.; et al. LIBRAS – Conhecimento além dos sinais. São Paulo: Pearson, 2011.
- PIMENTA, N.; QUADROS, R. M. Curso de LIBRAS 1 – Iniciante. 3.ª edição. Porto Alegre: Pallotti, 2008.

Bibliografia Complementar

- ALMEIDA, E. C.; DUARTE, P. M. Atividades ilustradas em sinais da Libras. São Paulo: Revinter, 2004.
- BRITO, L. F. Por uma gramática de língua de sinais. Rio de Janeiro: Tempo Brasileiro, 1995.
- CAPOVILLA, F. C.; RAPHAEL, W. D.; MAURÍCIO, A. C. L. Dicionário Enciclopédico Ilustrado Trilíngue da Língua de Sinais Brasileira, v 1 e 2. São Paulo: Editora da Universidade de São Paulo, 2010.

- CAPOVILLA, F. C.; RAPHAEL, W. D. (ed.). Enciclopédia da Língua de Sinais Brasileira. v. 1 e 2. São Paulo: EDUSP, 2004.
- GESSER, A. LIBRAS? Que língua é essa?: Crenças e preconceitos em torno da língua de sinais e da realidade surda. São Paulo: Parábola Editorial, 2009.
- QUADROS, R. M.; KARNOPP, L. Língua de sinais brasileira: estudos linguísticos. ARTMED: Porto Alegre, 2004.

Sistemas Distribuídos

Bibliografia básica

- ANDREWS, G. R. Foundations of multithreaded, parallel, and distributed programming. Addison-Wesley, 2000.
- COULOURIS, G. F. et al. Distributed Systems: Concepts and Design, 5th edition, Addison-Wesley, 2012.
- KUROSE, J. F.; ROSS, K. Redes de Computadores e a Internet, 6.ª edição. Pearson Education, 2013.

Bibliografia complementar

- BIRMAN, K. P. Reliable distributed systems: technologies, web services, and applications, New York: Springer, 2005. ISBN 0387215093.
- CLARK, M. et al. Web services business strategies and architectures. Expert Press, 2002. ISBN 1904284132.
- JOSUTTIS, N. M. SOA in practice. O'Reilly, 2007. ISBN 0596529554.
- LYNCH, N. A Distributed algorithms. M. Kaufmann, 1997.
- TANENBAUM, A.S.; STEEN, M. van. Distributed Systems: Principles and Paradigms. Prentice Hall, 2nd Edition, 2006.

Metodologia e Experimentação em Engenharia de Software

Bibliografia básica

- JURISTO, N. Basics of software engineering experimentation. Kluwer Academic Publishers, 2001. ISBN 079237990X.
- WOHLIN, C. et al. Experimentation in Software Engineering. 2nd edition. Springer, 2012.

- GETTINBY, G; GARDINER, W. P. Experimental design techniques in statistical practice: a practical software-based approach. Horwood Pub., 1998. ISBN 1898563357.

Bibliografia complementar

- WAZLAWICK, R. S. Metodologia de Pesquisa para Ciência da Computação. Editora Campus, 2009.
- WOHLIN, C. Experimentation in software engineering: an introduction. Kluwer Academic, 2000. ISBN 0792386825.
- POLYAK, B. T. Introduction to optimization. Optimization Software, 1987. ISBN 0911575146.
- HALL, E. M. Managing risk methods for software systems development. Addison-Wesley, 1998. ISBN 0201255928.
- QUALIDADE no setor de software brasileiro. Brasilia, D.F.: MCT, 1997.

Mercado e Economia de Software

Bibliografia básica

- TOCKEY, S. Return on Software: Maximizing the Return on Your Software Investment, Addison-Wesley, 2004.
- GRADY, S. O. The software paradox: the rise and fall of the commercial software market. O'Reilly, 2015.
- SALIM, C. S. Construindo planos de negócios: todos os passos necessários para planejar e desenvolver negócios de sucesso. 3.ª edição. Rio de Janeiro: Elsevier, 2005. ISBN 8535217363.

Bibliografia complementar

- BROOKS J., F. P. O mítico homem-mês: ensaios sobre engenharia de software. Elsevier, 2009. ISBN 9788535234879.
- ENGHOLM J. H. Engenharia de software na prática. São Paulo: Novatec, 2010. ISBN 9788575222171.
- PIMENTEL, L. O. A proteção jurídica da propriedade intelectual de software: noções básicas e temas relacionados. IEL, 2008. ISBN 9788587683045.
- ORRICO J, H. Pirataria de Software. MM Livros, 2004. ISBN 8590424219.

- LINS, B. F. E. et al. O mercado de software no Brasil: problemas institucionais e fiscais Brasília (DF): Câmara dos Deputados, Coordenação de Publicações, 2007. ISBN 9788573654998.

Disciplinas específicas de Engenharia de Software

Engenharia de Sistemas

Bibliografia básica

- SCHNEIDEWIND, N. Systems and Software Engineering with Applications. New York, NY: IEEE, 2009.
- INCOSE. Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, version 3.2.2. International Council on Systems Engineering (INCOSE), INCOSE-TP-2003-002-03.2.2, 2012.
- WIDRIG, D.; LEFFINGWELL, D. Managing software requirements: a unified approach Boston: Addison-Wesley, 2001. ISBN 0201615932.

Bibliografia complementar

- MADACHY, R. J. Systems Engineering Principles for Software Engineers (Chapman & Hall/CRC Innovations in Software Engineering and Software Development Series) 1st Edition, CRC Press, 2016.
- WILSON, W. E. Conceptos sobre ingenieria de sistemas Buenos Aires: Centro regional de Ayuda Tecnica, 1968. 254 p., il.
- SILVA FILHO, B. S.; NISE, N. S. Engenharia de sistemas de controle. 3.ª edição. LTC, 2002, ISBN 85-216-1301-6.
- MAFFEO, B. Engenharia de software e especificação de sistemas. Rio de Janeiro: Campus, 1992. 484 p. ISBN 8570017014.
- NASCIMENTO, J. B. Metodologias de desenvolvimento de sistemas. São Paulo: Erica, 1993, ISBN 8571941483.

Construção de Software

Bibliografia básica

- MCCONNELL, S. Code Complete: um guia prático para a construção de software 2.ª edição. Porto Alegre, RS: Bookman, 2005. ISBN 8536305045.
- GOODLIFFE, P. Como ser um programador melhor. Novatec, 2015. ISBN 978-85-7522-415-1.
- ORAM, A.; WILSON, G. Beautiful code. O'Reilly, 2007. ISBN 9780596510046.

Bibliografia complementar

- FOWLER, M. Refatoração: aperfeiçoando o projeto de código existente Porto Alegre: Bookman, 2004, ISBN 8536303956.
- IRVINE, K. R. C++ and object-oriented programming Upper Saddle River: Prentice-Hall, 1997. 526 p. Bibliografia e indice ISBN 0023598522 (broch.)
- BECK, K. Implementation patterns Upper Saddle River, NJ: Addison-Wesley, 2008. ISBN 0321413091.
- MARTIN, R. C. Clean code: a handbook of agile software craftsmanship. Prentice Hall, 2009. ISBN 0132350882.
- AGANS, D. J. Debugging the nine indispensable rules for finding even the most elusive software and hardware problems. AMACOM, 2002. ISBN 0814471684.

Modelagem de Software

Bibliografia básica

- EVANS, E. Domain-driven design: tackling complexity in the heart of software Boston, MA: Addison-Wesley, 2004, ISBN 0321125215.
- MCLAUGHLIN, B. D.; POLLICE, G.; WEST, D. Head First Object-Oriented Analysis & Design, O'Reilly, 2006.
- AMBLER, S. W. Agile modeling: effective practices for eXtreme programming and the unified process. Wiley, 2002. ISBN 0471202827.

Bibliografia complementar

- MILLETT, S.; TUNE, N. Patterns, Principles and Practices of Domain-Driven Design. Wrox, 2015.
- FOWLER, M. Analysis patternsreusable object models. Addison-Wesley, 1997. ISBN 0201895420.
- MELLOR, S. J; SHLAER, S. Análise de sistemas orientada para objetos. São Paulo: McGraw-Hill, 1990.
- KOWAL, J. A Behavior models: specifying user's expectations. Prentice Hall, 1992. ISBN 0132927152.
- WAGNER, F. Modeling software with finite state machinesa practical approach. Auerbach, 2006. ISBN 9780849380860

Processos de Software

Bibliografia Básica

- FAIRLEY, R. E. Managing and Leading Software Projects, Wiley-IEEE Computer Society Press, 2009.
- ISO/IEC/IEEE 12207:2008. Standard for Systems and Software Engineering-Software Life Cycle Processes. Portal: http://goo.gl/5ixAtq.
- LARMAN, C. Agile and iterative development: a manager's guide. Addison-Wesley, 2004. ISBN 9780131111554.

Bibliografia Complementar

- JOHNSON, K. A.; KULPA, M. K. Interpreting the CMMIa process improvement approach. Auerbach, 2003. ISBN 0849316545.
- BEEDLE, M.; SCHWABER, K. Agile software development with Scrum. Prentice Hall, 2002. ISBN 0130676349.
- PROWELL, S. J. Cleanroom software engineering technology and process Reading, Mass.: Addison-Wesley, 1999. ISBN 0201854805.
- CAPUTO, K. CMM implementation guide choreographing software process improvement. Addison-Wesley, 1998. ISBN 0201379384.

- MAGUIRE, S. Debugging the development processpractical strategies for staying focused, hitting ship dates, and building solid teams. Microsoft, 1994. ISBN 1556156502.

Domínios de Software

Bibliografia básica

- PILONE, D.; MILES, R.; Head-First Software Development, O'Reilly, 2008.
- MCLAUGHLIN, B. D.; POLLICE, G.; WEST, D. Head First Object-Oriented Analysis & Design, O'Reilly, 2006.
- BOOCH, G. Object-oriented analysis and design with applications. 2nd edition. Addison-Wesley, 1994. ISBN 0805353402.

Bibliografia complementar

- WAZLAWICK, R. S. Análise e design orientados a objetos para sistemas de informação: Modelagem com UML, OCL e IFML. 3.ª edição. Campus, 2010.
- PENKER, M.; ERIKSSON, H. E. Business modeling with UML: business patterns at work. John Wiley & Sons, 2000. ISBN 0471295515.
- ODELL, J. J. Advanced object-oriented analysis and design using UML. Cambridge University Press, SIGS Books, 1998. ISBN 052164819X
- COCKBURN, A. Agile software development Boston: Addison-Wesley, 2002. (Agile software development series). ISBN 0201699699.
- BEEDLE, M.; SCHWABER, K. Agile software development with Scrum. Prentice Hall, 2002. (Series in agile software development). ISBN 0130676349.

Processos de Qualidade de Software

Bibliografia Básica

- Stefan Wagner. Software Product Quality Control. Springer-Verlag Berlin Heidelberg. 2013.

- Software Quality Assurance: From Theory to Implementation, Daniel Galin, Addison-Wesley, 2004
- MURPHY, Mark L C/C++ software quality tools Upper Saddle River: Prentice Hall PTR, 1996. ISBN 0134451236.

Bibliografia Complementar

- OSCIANSK, A.; SOARES, M. S. Qualidade de Software. Editora Novatec, 2007.
- CERTIFICAÇÃO CERTICS: um instrumento de política pública para inovação tecnológica em software. CTI Renato Archer, 2015. ISBN 9788565163088.
- KELLIHER, T. P.; KEEGAN, J. G.; OLIVER, D. W. Engineering complex systems with models and objects. McGraw-Hill, 1997. ISBN 0070481881.
- KAN, S. H. Metrics and models in software quality engineering. 2nd edition. Addison-Wesley, 2003. ISBN 0201729156.
- BARTIÉ, A. Garantia da qualidade de software. Campus, 2002. ISBN 8535211245.

Gerência de Projeto de Software

Bibliografia básica

- PMI. Um Guia do Conhecimento em Gerenciamento de Projetos. 5.ª edição. São Paulo: Saraiva, 2012. ISBN 9788502223721.
- CHEMUTURI, M. K.; CAGLEY, T. M. Jr. Mastering Software Project Management: Best Practices, Tools and Techniques. J. Ross Publishing, 2010. ISBN-13: 978-1604270341.
- COHN, M. Agile estimating and planning Upper Saddle River, NJ: Prentice Hall Professional Technical Reference, 2006. ISBN 0131479415.

Bibliografia complementar

- FAIRLEY, R.E. Managing and Leading Software Projects. John Wiley & Sons, 2009.
- HIGHSMITH, J. A. Agile project management: creating innovative products 2nd. edition. Addison-Wesley, 2010. ISBN 0321658396.

- SCHWABER, K. Agile project management with Scrum. Microsoft Press, 2004. ISBN 073561993X.
- DEMARCO, T. Controle de projetos de software: gerenciamento, avaliação, estimativa. Campus, 1991. ISBN 8570015283.
- MARTINS, J. C. C. Gerenciando projetos de desenvolvimento de software com PMI, RUP e UML. 4.ª edição. Brasport, 2007.

Design de Software

Bibliografia básica

- BUDGEN, D. Software Design, International Edition, 2nd Edition, Pearson Education, 2011.
- HALL, G. M. Adptive Code via C#: Agile coding with design patterns and SOLID principles. Microsoft Press, 2014.
- RUPING, A. Agile documentationa pattern guide to producing lightweight documents for software projects. Wiley, 2003. ISBN 0470856173.

Bibliografia complementar

- MILLETT, S.; TUNE, N. Patterns, Principles, and Practices of Domain-Driven Design, John Wiley & Sons, 2015.
- ARGILA, C.; YOURDON, E. Análise e projeto orientados a objetos: estudos de casos São Paulo: Makron Books, 1999. ISBN 8534609756.
- LARMAN, C. Applyng UML and patternsan introduction to object-oriented analysis and design. Prentice Hall PTR, 1997. ISBN 0137488807.
- EVANS, E. Domain-driven designtackling complexity in the heart of software Boston, MA: Addison-Wesley, 2004. ISBN 0321125215.
- HORSTMANN, C. S. Object-oriented design & patterns. 2nd edition. Hoboken, NJ: Wiley, 2006. ISBN 0471744875.

Governança e Gestão de Serviços de Software

Bibliografia básica

- GREMBERGEN, W. V.; HAES, S. Enterprise Governance of Information Technology: Achieving Strategic Alignment and Value. Springer, 2010. ISBN-13 978-1441946621.
- BROOKS, P. Metrics for Service Management: Designing for ITIL. 1st edition. Van Haren Publishing, 2012.
- SHIAVON, M. Acordos de nível operacional para controle do processo de manutenção de software, Mestrado em Engenharia Elétrica, Universidade Federal de Goiás, 2009.

Bibliografia complementar

- GUTH, S. Contract Negotiation Handbook: Software as a Service. Guth Ventures LLC, 2013.
- ERDOGMUS, H.; TANIR, O. Advances in software engineering comprehension, evaluation, and evolution. Springer, 2002. ISBN 0387951091.
- HASS, A. M. J. Configuration management principles and practice. Addison-Wesley, 2003. (The Agile software development series). ISBN 0321117662.
- MARSHALL, C. Enterprise modeling with UML: designing successful software through business analysis. Addison-Wesley, 2000. ISBN 0201433133.
- CENTRO DE GESTÃO E ESTUDOS ESTRATÉGICOS. Ciência, tecnologia e sociedade: novos modelos de governança. Brasilia, CGEE, 2005.

Arquitetura de Software

Bibliografia Básica

- BASS, L.; CLEMENTS, P.; KAZMAN, R. Software Architecture in Practice, 3rd edition, Pearson Education, Inc., 2013.
- CLEMENTS, P.; et al. Documenting Software Architectures: Views and Beyond. 2nd edition, Pearson Eduction, 2011.

- MENDES, A. Arquitetura de software: desenvolvimento orientado para arquitetura. Rio de Janeiro: Campus, 2002. ISBN 853521013X.

Bibliografia Complementar

- CERVANTES, H.; KAZMAN, R. Designing Software Archtectures: A Practical Approach, ISBN-13: 978-0134390789, Addison-Wesley, 2016.
- PUTMAN, J. Architecting with RM-ODP. Prentice Hall, 2001. ISBN 0130191167.
- HEINEMAN, G. T.; COUNCILL, W. T. Component-based software engineering: putting the pieces together. Addison-Wesley, 2001. ISBN 0201704854.
- BOSCH, J. Design and use of software architectures: adopting and evolving a product-line approach. Addison-Wesley, 2000. ISBN 0201674947.
- JOHNSON, R. Expert one-on-one J2EE development without EJB. Wiley Publishing, 2004. ISBN 0764558315.

Requisitos de Software

Bibliografia Básica

- WIEGERS, K. E. Software Requirements. Microsoft Press, 3rd edition, 2013.
- ROBERTSON, S. Mastering the Requirements Process: Getting Requirements Right. Addison-Wesley Professional, 3rd edition, 2012.
- WINTERS, J. P.; SCHNEIDER, G. Applying use cases: a practical guide, 2nd edition. Addison-Wesley, 2001. (The Addison-Wesley object technology series). ISBN 0201708531.

Bibliografia Complementar

- COCKBURN, A. Writing Effective Use Cases. Addison-Wesley, 2000.
- MELLOR, S. J; SHLAER, S. Análise de sistemas orientada para objetos. São Paulo: McGraw-Hill, 1990.
- MILLER, G.; ARMOUR, F. Advanced use case modeling: software systems. Boston: Addison-Wesley, 2001. (Addison-Wesley object technology series) ISBN 0201615924.

- YOURDON, E.; COAD, P. Análise baseada em objetos. 2a. edição. Rio de Janeiro: Campus, 1996. ISBN 8535200428.
- WIEGERS, K. E. More about software requirements: thorny issues and practical advice. Microsoft Press, 2006. ISBN 9780735622678.

Experiência do Usuário de Software

Bibliografia básica

- GOTHELF, J.; SEIDEN, J. Lean UX: Applying Lean Principles to Improve User Experience, O'Reilly, 2013.
- KRUG, S., Don´t make me think: revisited. New Riders, 2014. ISBN 978-0321965516.
- BORBA, F. E. Ajax: guia de programação. São Paulo: Erica, 2006. ISBN 8536501375.

Bibliografia complementar

- PREECE, J.; SHARP, H.; ROGERS, Y. Interaction Design: Beyond Human-Computer Interaction, Wiley, 2015. ISBN 978-1119020752.
- CYBIS, W. O. Ergonomia e usabilidade: conhecimentos, métodos e aplicações. 2a. edição. São Paulo: Novatec, 2010. ISBN 9788575222324.
- COUTAZ, J.; BASS, L. Developing software for the user interface. Addison-Wesley, 1991. ISBN 0201510464.
- HARTSON, H. R.; HIX, D. Developing user interfaces: ensuring usability through product & process. Wiley, 1993. ISBN 0471578134.
- OLSEN, D. R. Developing user interfaces. Morgan Kaufmann, 1998. ISBN 1558604189.

Software Concorrente e Distribuído

Bibliografia básica

- BUTCHER, P. Seven Concurrent Models in Seven Weeks. Pragmatic Bookshelf, 2014. ISBN-10: 1937785653
- DAIGNEAU, R. Service Design Pattern. Addison-Wesley, 2011. ISBN 032154420X.

- FOSTER, I. Designing and building parallel programsconcepts and tools for parallel software engineering Reading, Mass.: Addison-Wesley, 1995. ISBN 0201575949.

Bibliografia complementar

- NEWMAN, S. Building Microservices. O'Reilly, 2015.
- BARNES, L. L. Client/server & beyound strategies for 21st century. Prentice-Hall, 1997. ISBN 0135325161.
- SHATZ, S. M. Development of distributed software concepts and tools. Macmillan, 1993. ISBN 0024096113.
- PAGE-JONES, M. Fundamentals of object-oriented design in UML. Addison-Wesley, 2003. ISBN 020169946X.
- RODRIGUES, L.; GUERRAOUI, R. Introduction to reliable distributed programming. Springer, 2006. ISBN 9783540288459.

Padrões de Arquitetura de Software

Bibliografia Básica

- FOWLER, M., Patterns of Enterprise Application Architecture, ISBN-13: 978-0321127426, Addison-Wesley, 2002.
- DAIGNEAU, R., Service Design Patterns, ISBN-13: 978-0321544209, Addison-Wesley, 2011.
- BUSCHMAN, F. Pattern-oriented software architecture. Wiley, 2001. ISBN 0471606952.

Bibliografia Complementar

- VERNON, V. Implementing Domain-Driven Design. Addison-Wesley, 2013. ISBN 978-0-321-83457-7.
- GAMMA, E. Design patterns elements of reusable object-oriented software. Addison Wesley, 1995. ISBN 0201633612.
- TROTT, J.; SHALLOWAY, A. Design patterns explained: a new perspective on object-oriented design. 2nd ed. Addison-Wesley, 2005. ISBN 0321247140.
- PREE, W. Design patterns for object-oriented software development. Addison-Wesley, 1995. ISBN 0201422948.

- FREEMAN, E. et al. Head First design patterns. O'Reilly, 2004. ISBN 0596007124.

Teste de Software

Bibliografia Básica

- MYERS, G. J. The Art of Software Testing. Wiley. 2011.
- COPELAND, L. A Practitioner's Guide to Software Test Design. Artech House, 2003.
- SYKES, D. A.; MCGREGOR, J. D. A practical guide to testing object-oriented software. Addison-Wesley, 2001. ISBN 0201325640.

Bibliografia Complementar

- MESZAROS, G. xUnit test patternsrefactoring test code. Addison-Wesley, 2007. ISBN 0131495054.
- MOSLEY, D. J. Client-server software testing on the desktop and the Web. Prentice Hall, 2000. ISBN 0131838806.
- GROSS, H. Component-based software testing with UML. Springer, 2005. ISBN 354020864X.
- PERRY, W. E. Effective methods for software testing. 2nd edition. J. Wiley, 1999. ISBN 047135418X.
- WU, M.; LI, K. Effective software test automationdeveloping an automated software testing tool. SYBEX, 2004. ISBN 0782143202.

Software para Sistemas Ubíquos

Bibliografia básica

- MCGRAW, G. Software Security: Build Security In. Addison-Wesley, 2006.
- GREENGARD, S. The Internet of Things. The MIT Press, 2015.
- WILMSHURST, T. An introduction to the design of small-scale embedded systems. Palgrave, 2001. ISBN 0333929942.

Bibliografia complementar

- MANICO, J.; DETLEFESEN, A. Iron-Clad Java: Building Secure Web Applications. McGraw-Hill, 2014.

- HOWARD, M. 19 deadly sins of software securityprogramming flaws and how to fix them. McGraw-Hill/Osborne, 2005. ISBN 0072260858.
- MARK, D. Dominando o desenvolvimento no iPhone: explorando o SDK do iPhone. Alta Books, 2009. ISBN 97885760833757.
- VAHID, F. Embedded system design: a unified hardware/software introduction. John Wiley & Sons, 2002. ISBN 0471386782.
- KERN, C.; KESAVAN, A.; DASWANI, N. Foundations of security: what every programmer needs to know. Apress, 2007. ISBN 9781590597842.

Software para Persistência de Dados

Bibliografia básica

- REDMOND, E.; WILSON, J. Seven databases in Seven Weeks. Pragmatic Bookshelf, 2012.
- MCMURTRY, D. et al. Data Access for Highly-Scalable Solutions: Using SQL, NoSQL, and Polyglot Persistence. Microsoft Press, 2013.
- AMBLER, S. W. Agile database techniques: effective strategies for the agile software developer. Wiley, 2003. ISBN 0471202835.

Bibliografia complementar

- AMBLER, S. J.; SADALAGE, P. J. Refactoring Databases. Addison-Wesley, 2006.
- GOLENDZINER, L. G.; PRICE, R. T. Bancos de dados para aplicações não convencionais. EBAI, 1989. ISBN 950139879.
- FALINO, J.; HERION, D.; MARTINER, W. Building distributed applications with ADO. John Wiley & Sons, 1999. ISBN 0471317012.
- KING, G.; BAUER, C. Java Persistence com Hibernate. Ciencia Moderna, 2007. ISBN 9788573936148.
- MANNINO, M. V. Projeto, desenvolvimento de aplicações e administração de banco de dados. McGraw-Hill, 2008. ISBN 9788577260201.

Prática em Engenharia de Software

Bibliografia básica

- SOMMERVILLE, I. Engenharia de Software. Pearson, 10.ª edição, 2015.

- MCCONNELL, S. Code complete: a practical handbook of software construction. 2nd edition. Microsoft Press, 2009.
- PAULA, W. P. Engenharia de software: fundamentos, métodos e padrões 3.ª edição. LTC, 2009. ISBN 9788521616504.

Bibliografia complementar

- PRESSMAN, R. Engenharia de Software. 8.ª edição. McGraw-Hill, 2014.
- SCHACH, S. R. Engenharia de software: os paradigmas clássicos & orientados a objetos. 7.ª edição. McGraw-Hill, 2009. ISBN 9788577260454.
- PFLEEGER, S. L. Engenharia de software: teoria e prática. 2.ª edição. Prentice Hall, 2004. ISBN 8587918311.
- PEDRYCZ, W.; PETERS, J. F. Engenharia de software: teoria e prática. Campus, 2003. ISBN 8535207465.
- MAGELA, R. Engenharia de software aplicada: princípios. Alta Books, 2006. ISBN 8576081202.