Universidade do Porto
FEUP Faculdade de Engenharia

Departamento de Engenharia Electrotécnica e de Computadores
Secção de Informática

Coordenação Curricular

Relatório de progresso

Versão 1.08, 18/07/2013, 11:14:46.

Índice por Sub-áreas

i. Introdução [JCL]

A Secção de Informática do DEEC em reunião de 13/2/2002 resolveu iniciar a discussão sobre a Coordenação Curricular na área da Informática, sem ultrapassar o âmbito das suas competências. Foram consideradas as disciplinas a cargo da Secção de Informática de acordo com a afectação de disciplinas a Secções do DEEC aprovada na reunião de 20/03/2002 do Conselho de Departamento do DEEC. Tendo ficado encarregado de lançar as ideias para a discussão, Eugénio Oliveira efectuou várias reuniões com Correia Lopes, Cristina Ribeiro e Pascoal Faria.
Posteriormente, a Secção de Informática do DEEC em reunião de 6/3/2002 nomeou uma equipa de trabalho com um responsável por cada sub-área da secção, encarregado de detalhar a informação sobre as disciplinas da sub-área, analisar as dependências com outras disciplinas da sub-área ou fora dela e produzir um conjunto de comentários e recomendações, e um coordenador encarregado de manter a informação em base de dados por forma a produzir relatórios com informação útil à Coordenação Curricular.

Esta é a última versão do relatório (18/07/2013, 11:14:46), 40 páginas numa fonte de tamanho 11pt e pode ser impressa a partir daqui.

i.1 Processo

O IEEE e a ACM fizeram sair muito recentemente o resultado de um trabalho de análise e de recomendações sobre os currículos no domínio da Informática, designado Computing Curricula 2001 on Computer Science (abreviado por CC2001). Considerou-se que qualquer trabalho de coordenação curricular deveria ter o CC2001 em atenção, não só por ser um trabalho profundo, abrangente e de grande aceitação internacional, mas também porque manter como base o CC2001 permite situar as componentes de informática dos cursos em que a Secção está envolvida num referencial bem conhecido.
O CC2001 divide as matérias a leccionar por 14 grandes Áreas de Conhecimento, estando estas por sua vez divididas em Unidades de Conhecimento, cada uma destas contendo um conjunto de Tópicos e de Objectivos de Aprendizagem.
No CC2001, algumas das unidades são marcadas como fundamentais (*core*) não devendo de forma alguma ser omitidas num curso de informática. No entanto, apenas o conjunto das unidades fundamentais, não forma matéria suficiente para um curso. Um curso deverá também abranger sempre uma parte significativa ou a totalidade das outras unidades

recomendadas (marcadas como *elective*). O presente trabalho procura identificar quais as Unidades e Tópicos do CC2001 (ou outros) que devem constar dos programas das disciplinas em que a secção de Informática está envolvida, bem como as relações e dependências com os programas de outras disciplinas, pertencentes a outras secções e departamentos. O nível da análise efectuada depende muito dos cursos: enquanto num curso como a LEIC se pretende garantir a cobertura de todas as áreas fundamentais da Informática, noutros cursos as propostas estão condicionadas pelo espaço dado á Informática nos planos de estudos. Uma disciplina poderá abranger unidades do CC2001 de mais do que uma área e também novas unidades não definidas no CC2001, se houver necessidade disso. Também é possível uma disciplina conter apenas parte dos tópicos de uma unidade CC2001. Nas situações em que as unidades do CC2001 não satisfaçam os requisitos das disciplinas dos cursos consideraram-se duas formas para obviar o problema. Em primeiro lugar acrescentaram-se novos tópicos a unidades já existentes completando-as em alguns aspectos que se consideraram em falta. Em segundo lugar criaram-se novas unidades (não CC2001) quando houve necessidade de cobrir todo um novo assunto contendo múltiplos tópicos não incluídos em unidades já definidas.

i.2 Base de Dados

Como pode ser observado no Esquema da Base de Dados (uma representação gráfica das tabelas e campos que constituem a Base de Dados e das associações entre tabelas), é guardada informação sobre as Disciplinas, as Áreas Científicas, as Secções e as suas Sub-áreas, os Docentes, a hierarquia do CC2001 (Áreas de Conhecimento, Unidades, Tópicos e Objectivos) e informação relativa ao conteúdo da Ficha de Disciplina, na forma de um associação de muitos para muitos entre Disciplinas e Tópicos e Disciplinas e Objectivos. Os Objectivos de Aprendizagem e os Tópicos a ministrar no curso encontram-se agrupados em Unidade de Conhecimento e estas em Áreas. Cada Área de Conhecimento está associada a uma Sub-área de Secção. É ainda registada a associação, de muitos para muitos, entre Disciplinas e Unidades de Conhecimento. A Base de Dados está já preparada para aceitar Pré-requisitos entre Unidades e o nível de profundidade a que cada tópico deve ser tratado.

i.3 Cursos

Este trabalho incide principalmente, como é natural, em disciplinas da LEIC, Licenciatura em Engenharia Informática e Computação, que concentra a maior fatia das disciplinas atribuídas à Secção. Este curso vai iniciar no próximo ano lectivo um novo plano de estudos resultante de uma Revisão Curricular iniciada em finais de 1998/1999. As disciplinas deste curso espalham-se pelas várias sub-áreas da secção mas apenas são tratadas as disciplinas da área científica de informática que foram atribuídas à secção. Algumas disciplinas que caberiam naturalmente neste análise ficam de fora porque, nesta altura, não pertencem à Secção de Informática. Considerou-se que o novo currículo da LEIC deveria cobrir, tanto quanto possível, o currículo recentemente recomendado pelo CC2001.

Relativamente à LEEC, Licenciatura em Engenharia Electrotécnica e de Computadores, as disciplinas da área da Informática têm sido globalmente consideradas como insuficientes para assegurar as competências pretendidas no perfil dos Electrotécnicos a formar. Para facilitar uma perspectiva global sobre este assunto, o diagrama de dependências para a LEEC na sub-área da Sistemas de Informação (a que tem maior peso) inclui todas as disciplinas de Informática. Pode concluir-se que, dependendo do ramo, os alunos têm entre 2 e 4 disciplinas obrigatórias de Informática e que existe uma grande concentração de optativas no 1º semestre do 5º ano, o que dificulta a utilização dos respectivos conhecimentos como ferramenta para outras disciplinas de Electrotecnia dando-lhes, pelo contrário, um carácter de especialização que à partida não teriam.

A LCI, Licenciatura em Ciência da Informação, foi criada em 2000/2001, na sequência da experiência adquirida no Mestrado em Gestão de Informação, dos contactos havidos com as associações profissionais da área das bibliotecas e arquivos, bem como com grupos de

investigação em várias universidades. O seu objectivo é preparar profissionais aptos a dirigirem serviços de documentação, arquivos, bibliotecas e serviços de informação em geral numa perspectiva integrada de gestão da informação das organizações, a qual actualmente só se compreende se solidamente baseada no conhecimento das tecnologias da informação.
A proposta surgiu do trabalho conjunto de elementos da FLUP e da FEUP e assume a forma de licenciatura conjunta das duas faculdades. Com um numerus clausus de 30 a 40 alunos, não é uma licenciatura de Informática, como o demonstra o facto de apenas cerca de 1/3 da carga docente corresponder à FEUP. Para um total de 37 disciplinas, incluindo estágio, a FEUP terá 12 das 32 disciplinas obrigatórias e 2 das 12 propostas de optativas.
As áreas científicas nucleares do curso [Total de 120 créditos] são:

1. Sistemas de Informação (SIST) [4 disciplinas; 12.5 UC]
2. Organização e Processamento de Informação (OPI) [4 disciplinas; 12 UC]
3. Serviços de Informação (SERV) [5 disciplinas; 14.5 UC]

e as áreas científicas complementares:

1. Ciências da Administração e da Gestão (CAG) [4 disciplinas; 14 UC]
2. Ciências Sociais e Humanas (CSH) [12 disciplinas; 34 UC]
3. Informática (I) [3 disciplinas; 9 UC].

No gráfico de dependências da LCI relativo à sub-área de Sistemas de Informação, apresentam-se todas as disciplinas que cabem à FEUP, mesmo as que são asseguradas por outras unidades da faculdade.

i.4 Estrutura do Relatório

Este documento, para além desta parte introdutória, contém partes para cada sub-área da secção (Arquitectura de Computadores e Sistemas Operativos, Ciência e Tecnologia da Programação, Engenharia de Software, Interacção e Multimédia, Sistemas de Informação, Sistemas Inteligentes) e uma parte final contendo relatórios, resultantes de interrogações à base de dados, que poderão ajudar a detectar inconsistências. Em cada parte, correspondente a uma sub-área, o trabalho é dividido por curso: LEIC, LEEC, LCI. Para cada curso é apresentado o conteúdo das várias disciplinas que o integram e que estão atribuídas à secção, um diagrama com as linhas de disciplinas e as dependências entre elas representadas por setas, um conjunto de comentários e recomendações e o conteúdo detalhado para cada disciplina.
Nos diagramas representando as dependências entre as disciplinas, agrupadas por linhas de disciplinas, foi seguida a seguinte notação: disciplinas da sub-área em análise estão em negrito; as disciplinas optativas são anotadas com (op); setas representando dependências fortes são representadas a traço mais grosso do que as setas representando dependências fracas; as linhas de disciplinas dentro da sub-área em análise aproximam-se das áreas de conhecimento do CC2001; nas sub-áreas da secção apenas foram consideradas disciplinas que, nesta altura, estão atribuídas à secção; a linha "Outros" acomoda as disciplinas que não pertencem a nenhuma sub-área da secção.

i.5 Sugestões e Correções

Em cada parte do documento encontra-se a indicação do respectivo responsável (por exemplo [JCL] nesta parte) para quem devem ser dirigidas todas as sugestões, as correcções e os comentários (por exemplo por email seguindo o link associado à indicação do responsável).

ii. Arquitectura de Computadores e Sistemas Operativos [APM]

ii.1 Introdução

Na Secção de Informática e no ano lectivo 2002/2003, esta sub-área abrange algumas disciplinas da área de Arquitectura de Computadores da LEIC, nomeadamente Arquitectura de Computadores (AC), Laboratório de Computadores (LC) e Arquitecturas Avançadas de Computadores (AAC), e ainda algumas disciplinas da área de Sistemas Operativos e Redes da LEIC, nomeadamente Sistemas Operativos (SO), Sistemas Distribuídos (SDist) e Computação Móvel (CM).

Na LEEC estão abrangidas duas disciplinas, Sistemas Operativos (SO) (área de Informática da LEEC) e Sistemas Distribuídos (SD) (área de Controlo Industrial da LEEC).

A Licenciatura em Ciência da Informação (LCI) inclui nesta área a disciplina de Sistemas Computacionais e da Comunicação (SCC).

Nos quadros seguintes resumem-se algumas das características destas disciplinas e a sua inserção nas respectivas áreas das licenciaturas.

Licenciatura em Engenharia Informática e Computação (LEIC):

Área de Arquitectura de Computadores	**Área de Sistemas Operativos e Redes**
Sistemas Digitais (1º ano - 1º sem.) (2T + 2TP)	**Sistemas Operativos** (3º ano - 1º sem.) (3T + 1TP)
Arquitectura de Computadores (1º ano - 2º sem.) (3T + 1TP)	**Sistemas Distribuídos** (3º ano - 2º sem.) (3T + 1TP)
Microprocessadores e Microcomputadores (2º ano - 2º sem.) (2T + 2TP)	Redes de Comunicação de Dados (4º ano - 1º sem.) (2T + 2TP)
Laboratório de Computadores (2º ano - 2º sem.) (1T + 3P)	Gestão de Redes (op) (4º ano - 2º sem.) (3T + 1TP)
Arquitecturas Avançadas de Computadores (op) (5º ano - 1º sem.) (3T + 1TP)	**Computação Móvel** (op) (5º ano - 1º sem.) (3T + 1TP)

Licenciatura em Engenharia Electrotécnica e de Computadores (LEEC):

Área de Informática (Ramo TEC)	**Área de Controlo Industrial (Ramo TEC)**
Sistemas Operativos (4º ano - 1º sem.) (3T + 2P)	**Sistemas Distribuídos** (op) (5º ano - 1º sem.) (3T + 1P)
Comunicação de Dados e Redes de Computadores I (4º ano - 2º sem.) (3T + 2P)	... (outras disciplinas da área)
... (outras disciplinas da área)	

As disciplinas anotadas com (op) são optativas e a negrito apresentam-se as disciplinas da Secção de Informática.

Nas disciplinas da área de Arquitectura de Computadores (LEIC) pretender-se-á leccionar numa primeira fase toda a constituição, funcionamento, e programação de baixo nível de sistemas de computação ao nível do computador pessoal, avançando-se mais tarde para arquitecturas multiprocessador e multicomputador.

Na área de Sistemas Operativos inicia-se o estudo da organização, algoritmos fundamentais, e programação, abrangendo concorrência, dos sistemas operativos de sistemas de computação individuais, avançando-se de seguida para os sistemas interligados e sua

exploração. Numa fase posterior abordam-se as arquitecturas e os problemas específicos dos sistemas de computação móveis.

ii.2 LEIC

ii.2.1 Conteúdo

AC Arquitectura de Computadores (1/2)
Overview of computer architecture. Machine level representation of data. Assembly level organization. Memory organization and architecture. I/O fundamentals. Functional organization.

LC Laboratório de Computadores (2/2)
Assembly level programming. Interfacing and communication. Development systems for low-level programming.

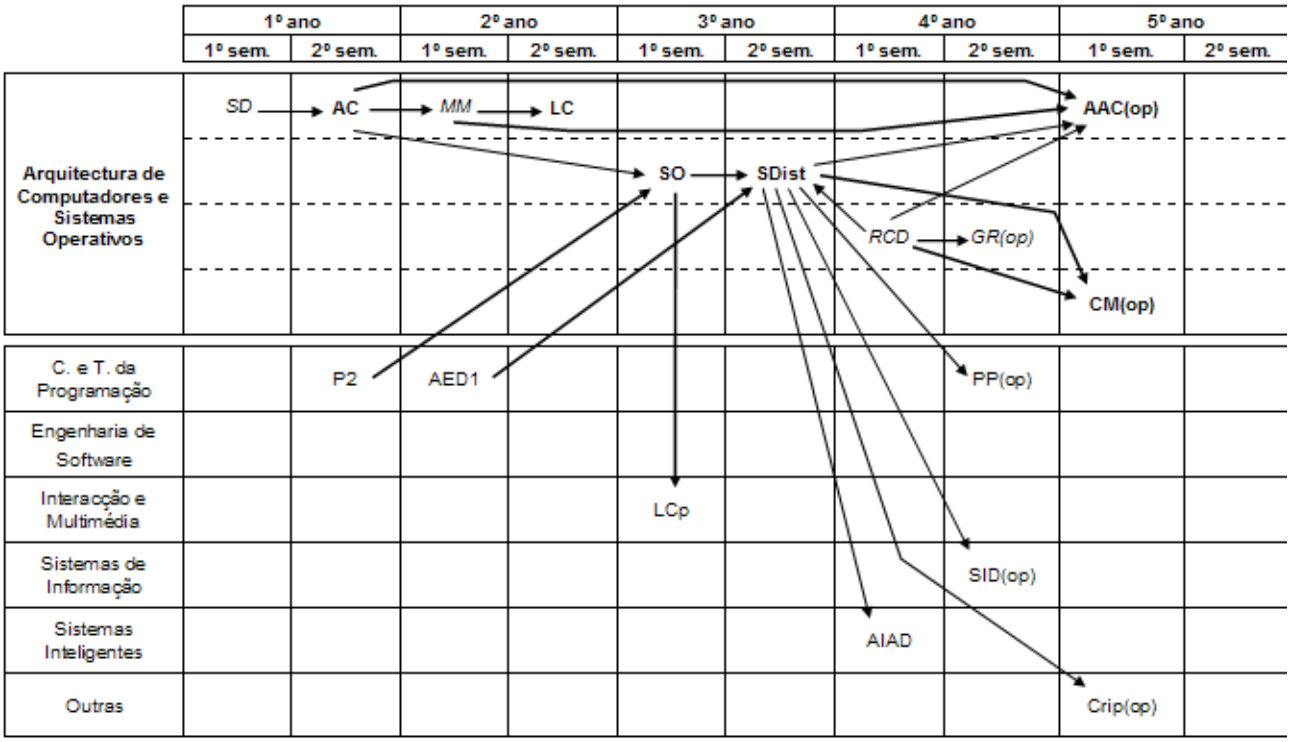
SO Sistemas Operativos (3/1)
Overview and the role of operating systems. Processes and concurrency. Scheduling and dispatch. Memory management. Device management and file systems. Security and protection.

Sdist Sistemas Distribuídos (3/2)
Net-centric computing and networks. Remote invocation and distributed objects. Naming and location. Replication and consistency. Fault tolerance. Security. Distributed infrastructures and platforms.

AAC Arquitecturas Avançadas de Computadores (5/1)
Multiprocessing and alternative computing. Architectures for networked and distributed systems. High performance architectures. Advanced interconnection technologies.

CM Computação Móvel (5/1)
Problems and specificities of wireless and mobile computing. Wireless networks. Mobile Internet protocols. Mobile data access. Support tools for mobile programming. Performance issues.

ii.2.2 Análise de Dependências



Análise das Dependências para Arquitectura de Computadores e Sistemas Operativos (LEIC)

ii.2.3 Comentários e Recomendações

I. Linha de Arquitectura de Computadores

SD Sistemas Digitais (1º ano - 1º semestre) (2T + 2TP)
Embora esta disciplina não pertença à Secção de Informática espera-se e sugere-se que cubra a maior parte da unidade AR01 do CC2001, com a possível exclusão do tópico 1, abordado na disciplina seguinte. Assim os tópicos a incluir da unidade AR01 - Digital logic and digital systems (core), são: Fundamental building blocks (logic gates, flip-flops, counters, registers, PLA); Logic expressions, minimization, sum of product forms; Register transfer notation; Physical considerations (gate delays, fan-in, fan-out). Obviamente, havendo tempo para isso, a disciplina poderá conter outros tópicos e unidades não CC2001.

AC Arquitectura de Computadores (1º ano - 2º semestre) (3T + 1TP)
Deverá cobrir com algum detalhe a arquitectura de von Neumann, incluindo as várias formas normalizadas de representação de dados. Deverá ainda incluir uma primeira introdução ligeira às instruções máquina, nomeadamente formatos e tipos de instruções, e na medida em que for necessário para a leccionação dos outros tópicos.

MM Microprocessadores e microcomputadores (2º ano - 2º semestre) (2T + 2TP)
Esta disciplina também não pertence à Secção de Informática, mas insere-se na sequência lógica de disciplinas da área de Arquitectura de Computadores. Espera-se que concretize o estudo dos tópicos da disciplina anterior para uma arquitectura actual ao nível do computador pessoal. A ênfase estaria no estudo do microprocessador, sua linguagem máquina mais detalhada, hardware para controlo do sistema de memória, e sistemas de entrada/saída. Estes últimos são apenas abordados de forma muito introdutória na disciplina anterior. Assim, sugere-se que o programa desta disciplina contenha os 5 últimos tópicos da unidade AR03 - Assembly level machine organization, estudados para um microprocessador concreto e actual; os 3 últimos tópicos de AR04 - Memory system organization and architectures, também concretizados; e ainda a unidade AR05 - Interfacing and communication.

LC Laboratório de Computadores (2º ano - 2º semestre) (1T + 3P)

Nesta disciplina ir-se-ão realizar trabalhos práticos relacionados com os temas leccionados principalmente nas 2 disciplinas anteriores, com maior incidência na última (MM). A disciplina de Sistemas Digitais inclui já os seus próprios trabalhos práticos. Assim são tópicos principais a prática da programação em assembly, explorando um processador actual; programação de dispositivos de entrada/saída ao nível do registo; a utilização de sistemas de desenvolvimento de baixo nível, possivelmente com uma linguagem compilada.

AAC Arquitecturas Avançadas de Computadores (5º ano - 1º semestre) (3T + 1TP) (opt.)
Pretende esta disciplina concluir o estudo das arquitecturas dos sistemas de computação, centrando-se principalmente nos sistemas multiprocessador e multicomputador e suas interligações.

Como se pode ver no diagrama apresentado em ii.2.2, as disciplinas desta área formam uma sequência lógica, que se traduz nas suas dependências, cobrindo todas as unidades do CC2001 referentes à área de Architecture and Organization (AR), com ligeiros acrescentos. As outras disciplinas da licenciatura terão poucas dependências destas directamente, para além do facto de que o conhecimento da organização e funcionamento interno dos sistemas de computação auxilia imenso todas as outras aprendizagens de tópicos da Ciência de Computadores. Pode contudo estabelecer-se uma dependência ligeira, mas mais directa, da disciplina de Sistemas Operativos relativamente a AC e MM, no que diz respeito a mecanismos de interrupção, execução priveligiada do processador, e controlo e gestão da memória.

II. Linhas de Sistemas Operativos

SO Sistemas Operativos (3º ano - 1º semestre) (3T + 1TP)
Esta disciplina lecciona os tópicos relativos à organização, funcionamento, algoritmos internos e programação dos sistemas operativos de um sistema de computação. Foca-se com alguma ênfase a concorrência de processos e threads e os mecanismos de sincronização. Outros tópicos incluem a gestão da memória, mecanismos de entrada/saída com ênfase nos sistemas de ficheiros, e ainda tópicos relativos à segurança. São cobertos, na sua grande maioria, os tópicos das unidades OS01 a OS08 do CC2001, o que não deixa de constituir um programa extenso. Diz a experiência que nos anos em que o número efectivo de semanas lectivas não atinge as 13,5/14, há grande dificuldade em leccionar todos os tópicos, nomeadamente os do último capítulo, que diz respeito à segurança. No entanto este último tópico poderá ser retomado na disciplina seguinte de Sistemas Distribuídos. Alguns dos trabalhos práticos relativos a tópicos desta disciplina poderão ocorrer na disciplina de Laboratório de Computação, que engloba também trabalhos de outra área.

SDist Sistemas Distribuídos (3º ano - 2º semestre) (3T + 1TP)
A disciplina de Sistemas Distribuídos lida tradicionalmente com vários temas relacionados com os sistemas de computação multicomputador interligados, que vão desde o seu sistema operativo até à exploração eficiente e correcta desse tipo de computação distribuída. Detalham-se temas que vão desde as bases da computação distribuída (como a programação de ligações de rede e a invocação remota de código) até às plataformas de desenvolvimento de mais alto nível de aplicações distribuídas. Abordam-se ainda outros tópicos como os algoritmos fundamentais do código distribuído, sincronização, sistemas de nomes, questões relacionadas com a replicação e consistência da informação, segurança, e tolerância a falhas, incluindo os mecanismos e protocolos do processamento transaccional. Os algoritmos distribuídos fundamentais poderão ser explicados à medida que forem necessários para o esclarecimento de outros temas, em vez de serem leccionados em capítulo próprio. Além de unidades do CC2001, das quais se seleccionaram os tópicos relevantes, achou-se útil introduzir mais 2 unidades englobando tópicos tradicionalmente leccionados na disciplina, mas não aparecendo de forma explícita em nenhuma outra unidade do CC2001. Devido à extensão do programa da disciplina alguns dos seus tópicos terão de ser abordados apenas a nível muito introdutório. Recomenda-se vivamente a troca de sequência desta disciplina com Redes e Comunicação de Dados, libertando-se assim SDist da necessidade de introduzir alguns tópicos de redes e sua programação (p. ex. sockets) na disciplina. A aprendizagem dos

tópicos de sistemas distribuídos beneficiaria bastante do conhecimento prévio mais aprofundado das redes de comunicações e seus protocos, serviços e programação. Na LEEC isso acontece. Do ponto de vista das dependências identificadas parece não haver grandes problemas.

RCD Redes de Comunicação de Dados (4º ano - 1º semestre) (2T + 2TP)
Esta disciplina não pertence à Secção de Informática mas porventura deverá ter uma coordenação relativamente forte com a disciplina de Sistemas Distribuídos. Deverá cobrir o grosso da unidade NC02 do CC2001, incluindo os protocolos mais usuais TCP/IP e UDP e a sua programação através da API quase universal de Sockets. Haveria toda a conveniência que precedesse a disciplina de Sistemas Distribuídos, como acontecia no currículo anterior.

CM Computação Móvel (5º ano - 1º semestre) (3T + 1TP) (opt.)
Em Computação Móvel abordam-se os problemas específicos e as tecnologias dos dispositivos de computação portátil e móvel na sua ligação e partilha de informação com os sistemas de computação interligados. A disciplina cobre a unidade NC09 - Wireless and mobile computing do CC2001 e pode considerar-se uma extensão quase directa das disciplinas anteriores de SDist e RCD.

A disciplina de Sistemas Operativos utiliza no seu modelo de programação o paradigma imperativo e a linguagem C, que é a interface habitual dos serviços de sistema, daí a dependência directa da disciplina onde se lecciona esse paradigma (Programação 2). Em Sistemas Distribuídos utiliza-se em geral o paradigma de orientação a objectos, daí a sua dependência de AED1. Na LEIC a disciplina de SDist precede a disciplina de base de redes, Redes de Comunicação de Dados, pelo que se torna necessário efectuar uma introdução prévia a esse tema, que permita a leccionação confortável dos outros tópicos da disciplina. No currículo anterior introduzia-se a programação de ligações em rede (API de sockets) na disciplina de Laboratório de Sistemas Operativos que agora desapareceu. Embora algum tempo da disciplina de Laboratório de Computação esteja ligado a Sistemas Operativos crê-se que já não seja possível introduzir esse tópico aí, que terá de transitar para SDist (ou ignorá-lo aí e transitar para RCD). As ligações de SO a PP derivam dos algoritmos paralelos utilizarem como base a programação concorrente. Pode ainda salientar-se alguma ligação entre SDist e Crip. Em SDist pretende-se leccionar algumas questões de segurança relativas aos sistemas interligados, mas apenas ao nível dos mecanismos e protocolos sem entrar em detalhes dos algoritmos efectivos de encriptação, que seriam explorados em profundidade na disciplina de Criptografia.

ii.2.4 Disciplinas

AC Arquitectura de Computadores (1/2)
Overview and history of computer architecture *[Digital logic and digital systems]*
Bits, bytes, and words *[Machine level representation of data]*
Numeric data representation and number bases *[Machine level representation of data]*
Fixed- and floating-point systems *[Machine level representation of data]*
Signed and twos-complement representations *[Machine level representation of data]*
Representation of nonnumeric data (character codes, graphical data) *[Machine level representation of data]*
Representation of records and arrays *[Machine level representation of data]*
Basic organization of the von Neumann machine *[Assembly level machine organization]*
Control unit; instruction fetch, decode, and execution *[Assembly level machine organization]*
Instruction sets and types (data manipulation, control, I/O) *[Assembly level machine organization]*
Instruction formats *[Assembly level machine organization]*
Addressing modes *[Assembly level machine organization]*
Subroutine call and return mechanisms *[Assembly level machine organization]*
Assembly/machine language programming *[Assembly level machine organization]*
Storage systems and their technology *[Memory system organization and architecture]*

Coding, data compression, and data integrity *[Memory system organization and architecture]*
Memory hierarchy *[Memory system organization and architecture]*
Main memory organization and operations *[Memory system organization and architecture]*
Latency, cycle time, bandwidth, and interleaving *[Memory system organization and architecture]*
Cache memories (address mapping, block size, replacement and store policy) *[Memory system organization and architecture]*
Virtual memory (page table, TLB) *[Memory system organization and architecture]*
Fault handling and reliability *[Memory system organization and architecture]*
I/O fundamentals: handshaking, buffering, programmed I/O, interrupt-driven I/O *[Interfacing and communication]*
Implementation of simple datapaths *[Functional Organization]*
Control unit: hardwired realization vs. microprogrammed realization *[Functional Organization]*
Instruction pipelining *[Functional Organization]*
Introduction to instruction-level parallelism (ILP) *[Functional Organization]*

LC Laboratório de Computadores (2/2)
Assembly/machine language programming *[Assembly level machine organization]*
Memory region organization in a program: Code, Data, Stack *[Assembly level machine organization]*
Low level operation of a microcomputer *[Assembly level machine organization]*
Subroutine call and return mechanisms *[Assembly level machine organization]*
Interrupt structures: vectored and prioritized, interrupt acknowledgment *[Interfacing and communication]*
Interfacing simpe peripherals: communication and other devices *[Interfacing and communication]*
Compiled languages for low-level development *[Development systems for low-level programming]*
Controlling the positioning of memory blocks: code, data, heap *[Development systems for low-level programming]*
Defining I/O addresses and I/O programming *[Development systems for low-level programming]*

SO Sistemas Operativos (3/1)
Role and purpose of the operating system *[Overview of operating systems]*
History of operating system development *[Overview of operating systems]*
Functionality of a typical operating system *[Overview of operating systems]*
Design issues (efficiency, robustness, flexibility, portability, security, compatibility) *[Overview of operating systems]*
Structuring methods (monolithic, layered, modular, micro-kernel models) *[Operating system principles]*
Abstractions, processes, and resources *[Operating system principles]*
Concepts of application program interfaces (APIs) *[Operating system principles]*
Application needs and the evolution of hardware/software techniques *[Operating system principles]*
Device organization *[Operating system principles]*
Interrupts: methods and implementations *[Operating system principles]*
Concept of user/system state and protection, transition to kernel mode *[Operating system principles]*
States and state diagrams *[Concurrency]*
Structures (ready list, process control blocks, and so forth) *[Concurrency]*
Dispatching and context switching *[Concurrency]*
The role of interrupts *[Concurrency]*
Concurrent execution: advantages and disadvantages *[Concurrency]*
The "mutual exclusion" problem and some solutions *[Concurrency]*
Deadlock: causes, conditions, prevention *[Concurrency]*
Models and mechanisms (semaphores, monitors, condition variables, rendezvous)

[Concurrency]
Producer-consumer problems and synchronization *[Concurrency]*
Preemptive and nonpreemptive scheduling *[Scheduling and dispatch]*
Schedulers and policies *[Scheduling and dispatch]*
Processes and threads *[Scheduling and dispatch]*
Review of physical memory and memory management hardware *[Memory management]*
Overlays, swapping, and partitions *[Memory management]*
Paging and segmentation *[Memory management]*
Placement and replacement policies *[Memory management]*
Working sets and thrashing *[Memory management]*
Caching *[Memory management]*
Abstracting device differences *[Device management]*
Buffering strategies *[Device management]*
Direct memory access *[Device management]*
Recovery from failures *[Device management]*
Files: data, metadata, operations, organization, buffering, sequential, nonsequential *[File systems]*
Directories: contents and structure *[File systems]*
Standard implementation techniques *[File systems]*
Overview of system security *[Security and protection]*
Policy/mechanism separation *[Security and protection]*
Security methods and devices *[Security and protection]*
Protection, access, and authentication *[Security and protection]*
Models of protection *[Security and protection]*
Memory protection *[Security and protection]*
Encryption *[Security and protection]*

Sdist Sistemas Distribuídos (3/2)
Network architectures *[Introduction to net-centric computing]*
The range of specializations within net-centric computing: Networks and protocols, Networked multimedia systems, Distributed computing, Mobile and wireless computing. *[Introduction to net-centric computing]*
Distributed and network OS's and middleware *[Introduction to net-centric computing]*
The client-server model *[Introduction to net-centric computing]*
Introduction to network protocols *[Communication and networking]*
Streams and datagrams *[Communication and networking]*
API's for network operations *[Communication and networking]*
Remote procedure calls (RPC) *[Building web applications]*
Lightweight distributed objects *[Building web applications]*
The role of middleware *[Building web applications]*
Support tools *[Building web applications]*
Naming entities *[Network management]*
Domain names and name services *[Network management]*
Loaction and discovery *[Network management]*
Consensus and election *[Distributed algorithms]*
Termination detection *[Distributed algorithms]*
Fault tolerance *[Distributed algorithms]*
Stabilization *[Distributed algorithms]*
Clocks and synchronization *[Replication and consistency]*
Caching, replication and inconsistency *[Replication and consistency]*
Consistency models and protocols *[Replication and consistency]*
Distribution protocols *[Replication and consistency]*
Fundamental concepts: reliable and available systems *[Fault tolerance]*
Spatial and temporal redundancy *[Fault tolerance]*
Methods used to implement fault tolerance *[Fault tolerance]*
Failure models and policies in distributed systems *[Fault tolerance]*
Distributed commit and transactions *[Fault tolerance]*

Failure recovery *[Fault tolerance]*
Authentication and access control problems in distributed systems *[Network security]*
Use of passwords and access control mechanisms *[Network management]*
Secret-key algorithms *[Network security]*
Public-key algorithms *[Network security]*
Authentication protocols *[Network security]*
Digital signatures *[Network security]*
Examples *[Network security]*
Distributed file systems (ex: NFS, AFS, Samba) *[Distributed infrastructure and platforms]*
Distributed shared memory *[Distributed infrastructure and platforms]*
Distributed object-based platforms (ex: CORBA, COM, Java RMI, .NET) *[Distributed infrastructure and platforms]*

AAC Arquitecturas Avançadas de Computadores (5/1)
Introduction to SIMD, MIMD, VLIW, EPIC *[Multiprocessing and alternative architectures]*
Systolic architecture *[Multiprocessing and alternative architectures]*
Interconnection networks (hypercube, shuffle-exchange, mesh, crossbar) *[Multiprocessing and alternative architectures]*
Shared memory systems *[Multiprocessing and alternative architectures]*
Cache coherence *[Multiprocessing and alternative architectures]*
Memory models and memory consistency *[Multiprocessing and alternative architectures]*
Superscalar architecture *[Performance enhancements]*
Branch prediction *[Performance enhancements]*
Prefetching *[Performance enhancements]*
Speculative execution *[Performance enhancements]*
Multithreading *[Performance enhancements]*
Scalability *[Performance enhancements]*
Impact of architectural issues on distributed algorithms *[Architecture for networks and distributed systems]*
Network computing *[Architecture for networks and distributed systems]*
Distributed multimedia *[Architecture for networks and distributed systems]*
Farms *[High Performance Distributed Architectures]*
Clusters *[High Performance Distributed Architectures]*
Advanced interconnection technologies *[High Performance Distributed Architectures]*

CM Computação Móvel (5/1)
Overview of the history, evolution, and compatibility of wireless standards *[Wireless and mobile computing]*
The special problems of wireless and mobile computing *[Wireless and mobile computing]*
Wireless local area networks and satellite-based networks *[Wireless and mobile computing]*
Wireless local loops *[Wireless and mobile computing]*
Mobile Internet protocol *[Wireless and mobile computing]*
Mobile aware adaption *[Wireless and mobile computing]*
Extending the client-server model to accommodate mobility *[Wireless and mobile computing]*
Mobile data access: server data dissemination and client cache management *[Wireless and mobile computing]*
Software package support for mobile and wireless computing *[Wireless and mobile computing]*
The role of middleware and support tools *[Wireless and mobile computing]*
Performance issues *[Wireless and mobile computing]*
Emerging technologies *[Wireless and mobile computing]*
Ciência e Tecnologia da Programação

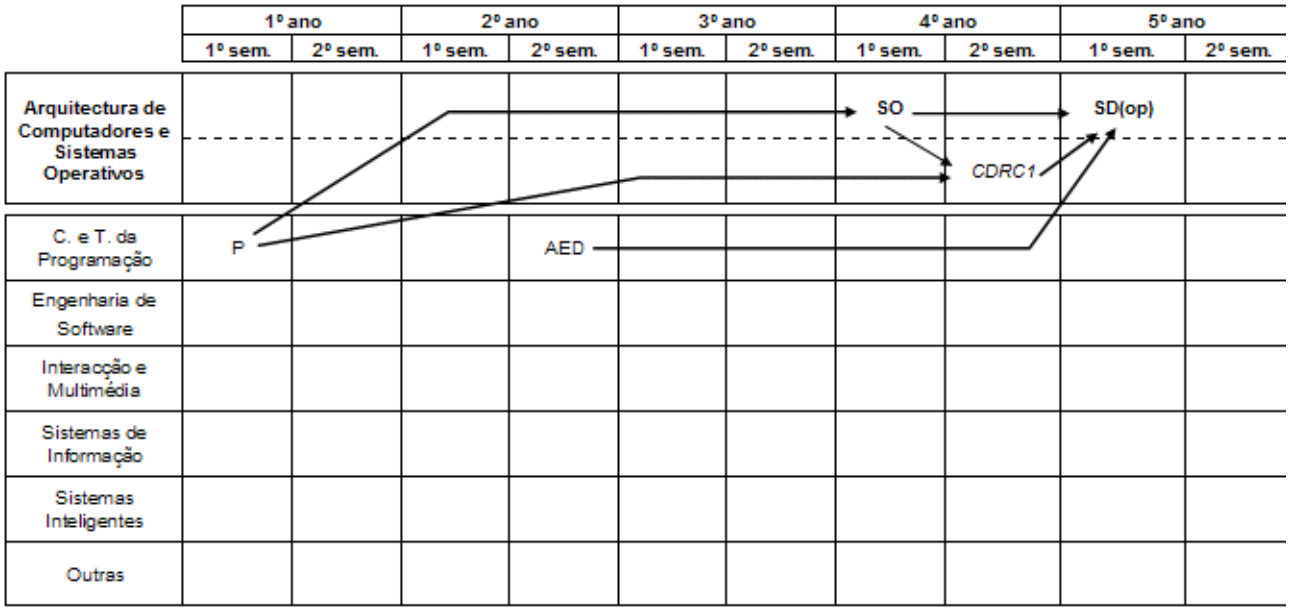
ii.3 LEEC

ii.3.1 Conteúdo

SO2 Sistemas Operativos (4/1)
Overview and the role of operating systems. Processes and concurrency. Scheduling and dispatch. Memory management. Device management and file systems. Security and protection.

SD Sistemas Distribuídos (5/1)
Net-centric computing. Remote invocation and distributed objects. Naming and location. Replication and consistency. Fault tolerance. Security. Distributed infrastructures and platforms.

ii.3.2 Análise de Dependências



Análise das Dependências para Arquitectura de Computadores e Sistemas Operativos (LEEC)

ii.3.3 Comentários e Recomendações

O que se disse relativamente às disciplinas de Sistemas Operativos e Sistemas Distribuídos da Licenciatura em Engenharia Informática e Computação (LEIC) aplica-se igualmente a esta licenciatura. A única diferença reside no facto da disciplina base de redes - Comunicação de Dados e Redes de Computadores 1 (CDRC1) - ser leccionada antes de Sistemas Distribuídos (que é de resto optativa nesta licenciatura). Em CDRC1 lecciona-se, entre outros assuntos, a constituição, protocolos e programação de ligações em rede (sockets), libertando Sistemas Distribuídos da necessidade de efectuar essa introdução.

ii.3.4 Disciplinas

SO2 Sistemas Operativos (4/1)
Role and purpose of the operating system *[Overview of operating systems]*
History of operating system development *[Overview of operating systems]*
Functionality of a typical operating system *[Overview of operating systems]*
Design issues (efficiency, robustness, flexibility, portability, security, compatibility) *[Overview of operating systems]*
Structuring methods (monolithic, layered, modular, micro-kernel models) *[Operating system principles]*
Abstractions, processes, and resources *[Operating system principles]*
Concepts of application program interfaces (APIs) *[Operating system principles]*
Device organization *[Operating system principles]*

Interrupts: methods and implementations *[Operating system principles]*
Concept of user/system state and protection, transition to kernel mode *[Operating system principles]*
States and state diagrams *[Concurrency]*
Structures (ready list, process control blocks, and so forth) *[Concurrency]*
Dispatching and context switching *[Concurrency]*
The role of interrupts *[Concurrency]*
Concurrent execution: advantages and disadvantages *[Concurrency]*
The "mutual exclusion" problem and some solutions *[Concurrency]*
Deadlock: causes, conditions, prevention *[Concurrency]*
Models and mechanisms (semaphores, monitors, condition variables, rendezvous) *[Concurrency]*
Producer-consumer problems and synchronization *[Concurrency]*
Preemptive and nonpreemptive scheduling *[Scheduling and dispatch]*
Schedulers and policies *[Scheduling and dispatch]*
Processes and threads *[Scheduling and dispatch]*
Review of physical memory and memory management hardware *[Memory management]*
Overlays, swapping, and partitions *[Memory management]*
Paging and segmentation *[Memory management]*
Placement and replacement policies *[Memory management]*
Working sets and thrashing *[Memory management]*
Caching *[Memory management]*
Abstracting device differences *[Device management]*
Buffering strategies *[Device management]*
Direct memory access *[Device management]*
Files: data, metadata, operations, organization, buffering, sequential, nonsequential *[File systems]*
Directories: contents and structure *[File systems]*
Standard implementation techniques *[File systems]*
Overview of system security *[Security and protection]*
Policy/mechanism separation *[Security and protection]*
Security methods and devices *[Security and protection]*
Protection, access, and authentication *[Security and protection]*
Memory protection *[Security and protection]*
Encryption *[Security and protection]*

SD Sistemas Distribuídos (5/1)
The range of specializations within net-centric computing: Networks and protocols, Networked multimedia systems, Distributed computing, Mobile and wireless computing. *[Introduction to net-centric computing]*
Distributed and network OS's and middleware *[Introduction to net-centric computing]*
The client-server model *[Introduction to net-centric computing]*
Remote procedure calls (RPC) *[Building web applications]*
Lightweight distributed objects *[Building web applications]*
The role of middleware *[Building web applications]*
Support tools *[Building web applications]*
Naming entities *[Network management]*
Domain names and name services *[Network management]*
Loaction and discovery *[Network management]*
Consensus and election *[Distributed algorithms]*
Termination detection *[Distributed algorithms]*
Fault tolerance *[Distributed algorithms]*
Stabilization *[Distributed algorithms]*
Clocks and synchronization *[Replication and consistency]*
Caching, replication and inconsistency *[Replication and consistency]*
Consistency models and protocols *[Replication and consistency]*
Distribution protocols *[Replication and consistency]*

Fundamental concepts: reliable and available systems *[Fault tolerance]*
Spatial and temporal redundancy *[Fault tolerance]*
Methods used to implement fault tolerance *[Fault tolerance]*
Failure models and policies in distributed systems *[Fault tolerance]*
Distributed commit and transactions *[Fault tolerance]*
Failure recovery *[Fault tolerance]*
Authentication and access control problems in distributed systems *[Network security]*
Use of passwords and access control mechanisms *[Network management]*
Secret-key algorithms *[Network security]*
Public-key algorithms *[Network security]*
Authentication protocols *[Network security]*
Digital signatures *[Network security]*
Examples *[Network security]*
Distributed file systems (ex: NFS, AFS, Samba) *[Distributed infrastructure and platforms]*
Distributed shared memory *[Distributed infrastructure and platforms]*
Distributed object-based platforms (ex: CORBA, COM, Java RMI, .NET) *[Distributed infrastructure and platforms]*

ii.4 LCI

ii.4.1 Conteúdo

SCC Sistemas Computacionais e de Comunicação (1/2)
Sistemas operativos. Sistema de ficheiros, gestão de processos, segurança, prática de UNIX e Windows. Redes de computadores: o modelo de referência OSI; protocolos. A família de protocolos TCP/IP. Serviços em redes IP: telnet, ftp, e-mail, nfs.

ii.4.2 Análise de Dependências

	1º ano		2º ano		3º ano		4º ano		5º ano	
	1º sem.	2º sem.	1º sem.	2º sem.	1º sem.	2º sem.	1º sem.	2º sem.	1º sem.	2º sem.
Arq. de Comp. e Sist. Operativos		SCC								
C. e T. da Programação										
Engenharia de Software										
Interacção e Multimédia										
Sistemas de Informação	IB									
Sistemas Inteligentes										
Outras										

Análise das Dependências para Arquitectura de Computadores e Sistemas Operativos (LCI)

ii.4.3. Comentários e recomendações

A disciplina de SCC (1º ano - 2º semestre) é uma das disciplinas base do conteúdo de informática desta Licenciatura. Nela se introduzem os conceitos de sistema de computação, os seus sistemas operativos, e a ligação de sistemas em rede com o estudo de alguns protocolos e serviços. Esta disciplina servirá de base para outras que se lhe seguem com conteúdos na área da informática.

ii.4.4 Disciplinas

SCC Sistemas Computacionais e de Comunicação (1/2)
Role and purpose of the operating system *[Overview of operating systems]*
History of operating system development *[Overview of operating systems]*
Functionality of a typical operating system *[Overview of operating systems]*
Mechanisms to support client-server models, hand-held devices *[Overview of operating systems]*
Design issues (efficiency, robustness, flexibility, portability, security, compatibility) *[Overview of operating systems]*
Influences of security, networking, multimedia, windows *[Overview of operating systems]*
Overview of system security *[Security and protection]*
Files: data, metadata, operations, organization, buffering, sequential, nonsequential *[File systems]*
Directories: contents and structure *[File systems]*
File systems: partitioning, mount/unmount, virtual file systems *[File systems]*
Fundamental concepts: reliable and available systems *[Fault tolerance]*
Basic system commands *[Scripting]*
Background and history of networking and the Internet *[Introduction to net-centric computing]*
Network architectures *[Introduction to net-centric computing]*
The range of specializations within net-centric computing: Networks and protocols, Networked multimedia systems, Distributed computing, Mobile and wireless computing. *[Introduction to net-centric computing]*
Distributed and network OS's and middleware *[Introduction to net-centric computing]*
The client-server model *[Introduction to net-centric computing]*
Network standards and standardization bodies *[Communication and networking]*
The ISO 7-layer reference model in general and its instantiation in TCP/IP *[Communication and networking]*
Circuit switching and packet switching *[Communication and networking]*
Fundamentals of cryptography *[Network security]*
Authentication protocols *[Network security]*
Digital signatures *[Network security]*

iii. Ciência e Tecnologia da Programação [MCR]

iii.1 Introdução

A subárea de Ciência e Tecnologia da Programação na Secção de Informática abrange as disciplinas das áreas de programação, algoritmos e linguagens de programação, bem como as de natureza mais teórica na área da teoria da computação, com tópicos como lógica, teoria de grafos, computabilidade e teoria da complexidade. Em termos do CC2001, nesta subárea concentram-se as disciplinas de 4 áreas (designação CC2001):

- Discrete Structures (DS)
- Programming Fundamentals (PF)
- Algorithms and Complexity (AL)
- Programming Languages (PL)

Esta cobertura mostra que esta subárea é responsável por um número apreciável de disciplinas nucleares da LEIC e das disciplinas de preparação em programação e algoritmos da LEEC.
As disciplinas da área CTP na LEIC cobrem uma área fundamental na licenciatura, centrada nos 1º e 2º anos, mas tendo também algum impacto nos seguintes tanto em disciplinas obrigatórias como optativas.

A organização das linhas de disciplinas da LEIC nesta subárea é baseada nas afinidades fortes

entre disciplinas, que sugerem a necessidade de 3 linhas: (1) Programação e algoritmos (2) Linguagens de Programação e (3) Teoria da Computação.
A organização de matérias nas disciplinas da linha (1) usou a experiência acumulada na licenciatura e as recomendações do CC2001. A introdução à programação é feita usando uma abordagem "primeiro funcional". Esta tem a vantagem de ser mais insensível à formação anterior dos alunos e de permitir uma introdução à programação numa linguagem compacta e com sintaxe leve: o Scheme. A abordagem "functional first" do CC2001 complementa a formação funcional com a formação em orientação aos objectos. Optou-se aqui por passar numa 2ª disciplina à abordagem imperativa. Desta forma é possível introduzir as construções de base das linguagens OO e não perder o paradigma imperativo necessário em diversas disciplinas de outras subáreas. O paradigma OO é incorporado como base das duas disciplinas de Algoritmos e Estruturas de Dados, que assim o podem usar nos trabalhos práticos. O paradigma lógico é objecto da disciplina de Programação em Lógica.
A linha (2) Linguagens de Programação tem apenas 1 disciplina obrigatória, Compiladores.
A linha (3) Teoria da Computação inclui as disciplinas de Lógica Computacional e Computabilidade e Linguagens Formais.

As disciplinas da área CTP na LEEC são Programação e Algoritmos e Estruturas de Dados. A abordagem nestas disciplinas é diferente da adoptada para a LEIC e centra-se nos paradigmas imperativo e OO. Havendo apenas 2 disciplinas na área da programação foi considerada essencial a competência nas linguagens C e C++, que são adoptadas nas 2 disciplinas.

iii.2 LEIC

iii.2.1 Conteúdo

P1 Programação I (1/1)
Elementos de programação funcional. Recursividade: procedimentos e processos. Iteração. Abstracção de dados. Construtores, selectores, modificadores. Algoritmos e resolução de problemas.

P2 Programação II (1/2)
Elementos de programação imperativa. Estrutura de programas. Tipos e declarações. Operadores e expressões. Entrada/saída. Funções e parâmetros. Estratégias de resolução de problemas. Depuração de programas.

AED1 Algoritmos e Estruturas de Dados I (2/1)
Programação Orientada por Objectos. Classes, hierarquias, herança, polimorfismo. Estruturas de dados lineares: uso de bibliotecas de estruturas de dados. Algoritmos de pesquisa. Algoritmos de ordenação. Eficiência de algoritmos. Classes de complexidade.

AED2 Algoritmos e Estruturas de Dados II (2/2)
Programação Orientada por Objectos. Programação por eventos. Árvores. Árvores equilibradas. Grafos. Algoritmos em grafos. Desenho de Algoritmos. Análise de Algoritmos. A classe de complexidade NP.

LoC Lógica Computacional (2/2)
Lógica proposicional e lógica de 1ª ordem. Métodos de prova. Provas formais. Teoria de conjuntos. Resolução. Semântica da Lógica de 1ª Ordem. Lógicas não clássicas.

PL Programação em Lógica (3/1)
Modelo de computação da programação em lógica. Resolução. Árvores de prova. Construções básicas. Linguagem Prolog. Estruturas de dados. Programação não-determinista. Meta-interpretadores.

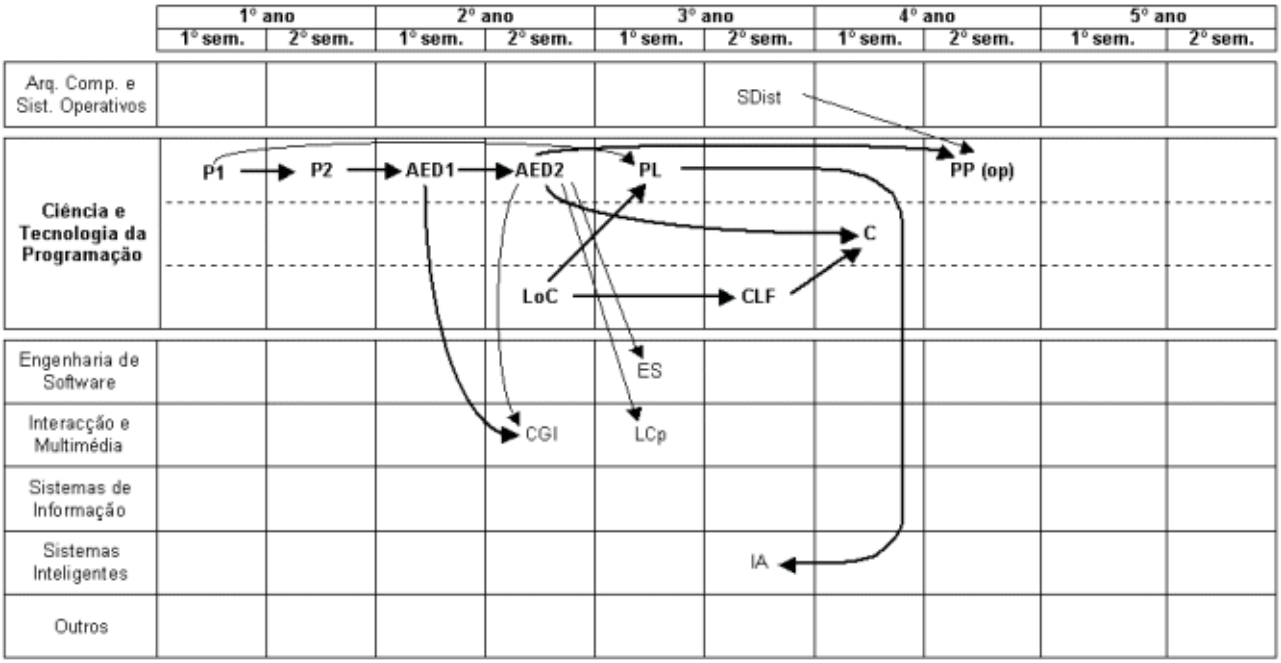
CLF Computabilidade e Linguagens Formais (3/2)
Autómatos finitos determinísticos. Linguagens regulares. Autómatos finitos

não-determinísticos. Gramáticas regulares. Expressões regulares. Gramáticas livres de contexto. Autómatos de pilha. Máquinas de Turing.

C Compiladores (4/1)
Tradução de linguagens de programação. Análise léxica, análise sintáctica, geração de código e optimização. Sistemas de tradução. Tipos e sua verificação. Semântica das linguagens de programação.

PP Programação Paralela (4/2)
Paralelismo e computação. Máquinas e arquitecturas paralelas. Paralelização de algoritmos. Programação de multi-computadores. Avaliação de sistemas paralelos. Programação de alto desempenho.

iii.2.2 Análise de Dependências



Análise das Dependências para Ciência e Tecnologia da Programação (LEIC)

iii.2.3 Comentários e Recomendações

Na LEIC vai entrar em vigor em 2002/2003 um novo plano de estudos, resultado de uma revisão que teve em conta as recomendações do CC2001 à data em que foi elaborado. É por isso natural que a organização das disciplinas desta subárea permita responder aos requisitos do núcleo de formação de uma licenciatura em Informática.

A área das linguagens de programação não é visível no plano de estudos, uma vez que as disciplinas que as tratam são as de Programação e de Algoritmos. É de recomendar a introdução de uma disciplina de linguagens de programação, de carácter optativo, para complementar a formação na área da integração de paradigmas, desenho de linguagens e sua semântica.

iii.2.4 Disciplinas

P1 Programação I (1/1)
Basic syntax and semantics of a higher-level language *[Fundamental programming constructs]*
Functions and parameter passing *[Fundamental programming constructs]*
Conditional and iterative control structures *[Fundamental programming constructs]*

Simple I/O *[Fundamental programming constructs]*
The concept of recursion *[Recursion]*
Recursive mathematical functions *[Recursion]*
Simple recursive procedures *[Recursion]*
The role of algorithms in the problem-solving process *[Algorithms and problem-solving]*
Implementation strategies for algorithms *[Algorithms and problem-solving]*
Debugging strategies *[Algorithms and problem-solving]*
Primitive types *[Fundamental data structures]*
Data representation in memory *[Fundamental data structures]*
Static, stack, and heap allocation *[Fundamental data structures]*
Linked structures *[Fundamental data structures]*
Recursive backtracking *[Recursion]*
Divide-and-conquer strategies *[Recursion]*
Implementation of recursion *[Recursion]*
Overview and motivation of functional languages *[Functional programming]*
Recursion over lists, natural numbers, trees, and other recursively-defined data *[Functional programming]*
Amortized efficiency for functional data structures *[Functional programming]*
Closures and uses of functions as data (infinite sets, streams) *[Functional programming]*
Pragmatics (debugging by divide and conquer; persistency of data structures) *[Functional programming]*
Backtracking *[Algorithmic strategies]*
Divide-and-conquer *[Algorithmic strategies]*
Greedy algorithms *[Algorithmic strategies]*
Simple numerical algorithms *[Fundamental computing algorithms]*
Sequential and binary search algorithms *[Fundamental computing algorithms]*
Quadratic sorting algorithms (selection, insertion) *[Fundamental computing algorithms]*
Using recurrence relations to analyze recursive algorithms *[Basic algorithmic analysis]*
Asymptotic analysis of upper and average complexity bounds *[Basic algorithmic analysis]*
History of programming languages *[Overview of programming languages]*
Brief survey of programming paradigms: Procedural languages, Object-oriented languages, Functional languages, Declarative, non-algorithmic languages, Scripting languages. *[Overview of programming languages]*

P2 Programação II (1/2)
History of programming languages *[Overview of programming languages]*
Brief survey of programming paradigms: Procedural languages, Object-oriented languages, Functional languages, Declarative, non-algorithmic languages, Scripting languages. *[Overview of programming languages]*
Basic syntax and semantics of a higher-level language *[Fundamental programming constructs]*
Variables, types, expressions, and assignment *[Fundamental programming constructs]*
Simple I/O *[Fundamental programming constructs]*
Conditional and iterative control structures *[Fundamental programming constructs]*
Functions and parameter passing *[Fundamental programming constructs]*
Structured decomposition *[Fundamental programming constructs]*
Problem-solving strategies *[Algorithms and problem-solving]*
The role of algorithms in the problem-solving process *[Algorithms and problem-solving]*
Implementation strategies for algorithms *[Algorithms and problem-solving]*
Debugging strategies *[Algorithms and problem-solving]*
The concept and properties of algorithms *[Algorithms and problem-solving]*
Primitive types *[Fundamental data structures]*
Arrays *[Fundamental data structures]*
Records *[Fundamental data structures]*
Strings and string processing *[Fundamental data structures]*
Data representation in memory *[Fundamental data structures]*
Static, stack, and heap allocation *[Fundamental data structures]*

Runtime storage management *[Fundamental data structures]*
Pointers and references *[Fundamental data structures]*
Brute-force algorithms *[Algorithmic strategies]*
Greedy algorithms *[Algorithmic strategies]*
Divide-and-conquer *[Algorithmic strategies]*
Simple numerical algorithms *[Fundamental computing algorithms]*
Sequential and binary search algorithms *[Fundamental computing algorithms]*
Quadratic sorting algorithms (selection, insertion) *[Fundamental computing algorithms]*
Declaration models (binding, visibility, scope, and lifetime) *[Declarations and types]*
Garbage collection *[Declarations and types]*
Identifying differences among best, average, and worst case behaviors *[Basic algorithmic analysis]*

AED1 Algoritmos e Estruturas de Dados I (2/1)
Object-oriented design *[Object-oriented programming]*
Encapsulation and information-hiding *[Object-oriented programming]*
Separation of behavior and implementation *[Object-oriented programming]*
Classes and subclasses *[Object-oriented programming]*
Inheritance (overriding, dynamic dispatch) *[Object-oriented programming]*
Polymorphism (subtype polymorphism vs. inheritance) *[Object-oriented programming]*
Class hierarchies *[Object-oriented programming]*
Primitive types *[Fundamental data structures]*
Linked structures *[Fundamental data structures]*
Implementation strategies for stacks, queues, and hash tables *[Fundamental data structures]*
Strategies for choosing the right data structure *[Fundamental data structures]*
The conception of types as a set of values with together with a set of operations *[Declarations and types]*
Declaration models (binding, visibility, scope, and lifetime) *[Declarations and types]*
Overview of type-checking *[Declarations and types]*
Garbage collection *[Declarations and types]*
Procedures, functions, and iterators as abstraction mechanisms *[Abstraction mechanisms]*
Parameterization mechanisms (reference vs. value) *[Abstraction mechanisms]*
Activation records and storage management *[Abstraction mechanisms]*
Type parameters and parameterized types *[Abstraction mechanisms]*
Modules in programming languages *[Abstraction mechanisms]*
Collection classes and iteration protocols *[Object-oriented programming]*
Internal representations of objects and method tables *[Object-oriented programming]*
O(N log N) sorting algorithms (Quicksort, heapsort, mergesort) *[Fundamental computing algorithms]*
Hash tables, including collision-avoidance strategies *[Fundamental computing algorithms]*
Binary search trees *[Fundamental computing algorithms]*
Asymptotic analysis of upper and average complexity bounds *[Basic algorithmic analysis]*
Identifying differences among best, average, and worst case behaviors *[Basic algorithmic analysis]*
Big O, little o, omega, and theta notation *[Basic algorithmic analysis]*
Standard complexity classes *[Basic algorithmic analysis]*
Empirical measurements of performance *[Basic algorithmic analysis]*
Time and space tradeoffs in algorithms *[Basic algorithmic analysis]*
Using recurrence relations to analyze recursive algorithms *[Basic algorithmic analysis]*

AED2 Algoritmos e Estruturas de Dados II (2/2)
Event-handling methods *[Event-driven programming]*
Event propagation *[Event-driven programming]*
Exception handling *[Event-driven programming]*
Trees *[Graphs and trees]*
Undirected graphs *[Graphs and trees]*
Directed graphs *[Graphs and trees]*

Spanning trees *[Graphs and trees]*
Traversal strategies *[Graphs and trees]*
Implementation strategies for graphs and trees *[Fundamental data structures]*
Strategies for choosing the right data structure *[Fundamental data structures]*
Binary search trees *[Fundamental computing algorithms]*
Representations of graphs (adjacency list, adjacency matrix) *[Fundamental computing algorithms]*
Depth- and breadth-first traversals *[Fundamental computing algorithms]*
Shortest-path algorithms (Dijkstra's and Floyd's algorithms) *[Fundamental computing algorithms]*
Transitive closure (Floyd's algorithm) *[Fundamental computing algorithms]*
Minimum spanning tree (Prim's and Kruskal's algorithms) *[Fundamental computing algorithms]*
Topological sort *[Fundamental computing algorithms]*
Greedy algorithms *[Algorithmic strategies]*
Divide-and-conquer *[Algorithmic strategies]*
Backtracking *[Algorithmic strategies]*
Branch-and-bound *[Algorithmic strategies]*
Heuristics *[Algorithmic strategies]*
Pattern matching and string/text algorithms *[Algorithmic strategies]*
Numerical approximation algorithms *[Algorithmic strategies]*
Definition of the classes P and NP *[The complexity classes P and NP]*
NP-completeness (Cook's theorem) *[The complexity classes P and NP]*
Standard NP-complete problems *[The complexity classes P and NP]*
Reduction techniques *[The complexity classes P and NP]*
Amortized analysis *[Advanced algorithmic analysis]*
Online and offline algorithms *[Advanced algorithmic analysis]*
Randomized algorithms *[Advanced algorithmic analysis]*
Dynamic programming *[Advanced algorithmic analysis]*
Combinatorial optimization *[Advanced algorithmic analysis]*

LoC Lógica Computacional (2/2)
Propositional logic *[Basic logic]*
Logical connectives *[Basic logic]*
Truth tables *[Basic logic]*
Normal forms (conjunctive and disjunctive) *[Basic logic]*
Validity *[Basic logic]*
Predicate logic *[Basic logic]*
Universal and existential quantification *[Basic logic]*
Modus ponens and modus tollens *[Basic logic]*
Limitations of predicate logic *[Basic logic]*
Notions of implication, converse, inverse, contrapositive, negation, and contradiction *[Proof techniques]*
The structure of formal proofs *[Proof techniques]*
Direct proofs *[Proof techniques]*
Proof by counterexample *[Proof techniques]*
Proof by contraposition *[Proof techniques]*
Proof by contradiction *[Proof techniques]*
Mathematical induction *[Proof techniques]*
Strong induction *[Proof techniques]*
Recursive mathematical definitions *[Proof techniques]*
Well orderings *[Proof techniques]*

PL Programação em Lógica (3/1)
Theory of logic programs *[Logic programming]*
Computational model for logic programming *[Logic programming]*
The resolution principle *[Logic programming]*

Unification algorithm *[Logic programming]*
Proof trees *[Logic programming]*
Strategy of the Prolog language *[Logic programming]*
Data structures *[Logic programming]*
Meta- interpreters *[Logic programming]*

CLF Computabilidade e Linguagens Formais (3/2)
Finite-state machines *[Basic computability]*
Context-free grammars *[Basic computability]*
Tractable and intractable problems *[Basic computability]*
Uncomputable functions *[Basic computability]*
The halting problem *[Basic computability]*
Implications of uncomputability *[Basic computability]*
Deterministic finite automata (DFAs) *[Automata theory]*
Nondeterministic finite automata (NFAs) *[Automata theory]*
Equivalence of DFAs and NFAs *[Automata theory]*
Regular expressions *[Automata theory]*
The pumping lemma for regular expressions *[Automata theory]*
Push-down automata (PDAs) *[Automata theory]*
Relationship of PDAs and context-free grammars *[Automata theory]*
Properties of context-free grammars *[Automata theory]*
Turing machines *[Automata theory]*
Nondeterministic Turing machines *[Automata theory]*
Sets and languages *[Automata theory]*
Chomsky hierarchy *[Automata theory]*
The Church-Turing thesis *[Automata theory]*

C Compiladores (4/1)
Comparison of interpreters and compilers *[Introduction to language translation]*
Language translation phases (lexical analysis, parsing, code generation, optimization) *[Introduction to language translation]*
Machine-dependent and machine-independent aspects of translation *[Introduction to language translation]*
Application of regular expressions in lexical scanners *[Language translation systems]*
Parsing (concrete and abstract syntax, abstract syntax trees) *[Language translation systems]*
Application of context-free grammars in table-driven and recursive-descent parsing *[Language translation systems]*
Symbol table management *[Language translation systems]*
Code generation by tree walking *[Language translation systems]*
Architecture-specific operations: instruction selection and register allocation *[Language translation systems]*
Optimization techniques *[Language translation systems]*
The use of tools in support of the translation process and the advantages thereof *[Language translation systems]*
Program libraries and separate compilation *[Language translation systems]*
Building syntax-directed tools *[Language translation systems]*
Data type as set of values with set of operations *[Type systems]*
Data types: Elementary types, Product and coproduct types, Algebraic types, Recursive types, Arrow (function) types, Parameterized types. *[Type systems]*
Type-checking models *[Type systems]*
Semantic models of user-defined types: Type abbreviations, Abstract data types, Type equality. *[Type systems]*
Parametric polymorphism *[Type systems]*
Subtype polymorphism *[Type systems]*
Type-checking algorithms *[Type systems]*
Informal semantics *[Programming language semantics]*
Overview of formal semantics *[Programming language semantics]*

Procedures, functions, and iterators as abstraction mechanisms *[Abstraction mechanisms]*
Parameterization mechanisms (reference vs. value) *[Abstraction mechanisms]*
Activation records and storage management *[Abstraction mechanisms]*

PP Programação Paralela (4/2)
Introduction to high-performance computing: History and importance of computational science, Overview of application areas, Review of required skills *[High-performance computing]*
High-performance computing: Processor architectures, Memory systems for high performance, Input/output devices, Pipelining, Parallel languages and architectures *[High-performance computing]*
Scientific visualization: Presentation of results, Data formats, Visualization tools and packages *[High-performance computing]*
Introduction to SIMD, MIMD, VLIW, EPIC *[Multiprocessing and alternative architectures]*
Interconnection networks (hypercube, shuffle-exchange, mesh, crossbar) *[Multiprocessing and alternative architectures]*
Shared memory systems *[Multiprocessing and alternative architectures]*
Superscalar architecture *[Performance enhancements]*
PRAM model *[Parallel algorithms]*
Algorithm parallelization techniques *[Parallel algorithms]*
Brent's theorem and work efficiency *[Parallel algorithms]*
Scalability *[Performance enhancements]*
Multithreading *[Performance enhancements]*
Network computing *[Architecture for networks and distributed systems]*
Schedulers and policies *[Scheduling and dispatch]*

iii.3 LEEC

iii.3.1 Conteúdo

P Programação (1/1)
Elementos de programação imperativa. Estrutura de programas. Tipos e declarações. Operadores e expressões. Entrada/saída. Funções e parâmetros. Estratégias de resolução de problemas. Depuração de programas.

AED Algoritmos e Estruturas de Dados (2/2)
Programação Orientada por Objectos. Classes, hierarquias, herança, polimorfismo. Estruturas de dados lineares: uso de bibliotecas de estruturas de dados. Algoritmos de pesquisa. Algoritmos de ordenação. Eficiência de algoritmos. Classes de complexidade.

iii.3.2 Análise de Dependências

	1º ano		2º ano		3º ano		4º ano		5º ano	
	1º sem.	2º sem.	1º sem.	2º sem.	1º sem.	2º sem.	1º sem.	2º sem.	1º sem.	2º sem.
Arq. Comp. e Sist. Operativos										
Ciência e Tecnologia da Programação	P →			AED						
Engenharia de Software										
Interacção e Multimédia										
Sistemas de Informação										
Sistemas Inteligentes										
Outros										

Análise das Dependências para Ciência e Tecnologia da Programação (LEEC)

iii.3.3 Comentários e Recomendações

A preparação em Programação dos alunos da LEEC tem sido considerada insuficiente pelos docentes das disciplinas de especialidade nos vários ramos. Este facto deveria ser tomado em atenção em futuras revisões do plano de estudos, onde seria interessante incluir projectos com componente de software. Nesta licenciatura torna-se particularmente importante, ao oferecer novas disciplinas da área da programação, articulá-las com as disciplinas das especialidades.

iii.3.4 Disciplinas

P Programação (1/1)
Brief survey of programming paradigms: Procedural languages, Object-oriented languages, Functional languages, Declarative, non-algorithmic languages, Scripting languages. *[Overview of programming languages]*
Basic syntax and semantics of a higher-level language *[Fundamental programming constructs]*
Variables, types, expressions, and assignment *[Fundamental programming constructs]*
Simple I/O *[Fundamental programming constructs]*
Conditional and iterative control structures *[Fundamental programming constructs]*
Functions and parameter passing *[Fundamental programming constructs]*
Structured decomposition *[Fundamental programming constructs]*
Problem-solving strategies *[Algorithms and problem-solving]*
The role of algorithms in the problem-solving process *[Algorithms and problem-solving]*
Implementation strategies for algorithms *[Algorithms and problem-solving]*
Debugging strategies *[Algorithms and problem-solving]*
The concept and properties of algorithms *[Algorithms and problem-solving]*
Primitive types *[Fundamental data structures]*
Arrays *[Fundamental data structures]*
Records *[Fundamental data structures]*
Strings and string processing *[Fundamental data structures]*
Data representation in memory *[Fundamental data structures]*
Static, stack, and heap allocation *[Fundamental data structures]*
Pointers and references *[Fundamental data structures]*
Brute-force algorithms *[Algorithmic strategies]*
Greedy algorithms *[Algorithmic strategies]*
Divide-and-conquer *[Algorithmic strategies]*

Simple numerical algorithms *[Fundamental computing algorithms]*
Sequential and binary search algorithms *[Fundamental computing algorithms]*
Quadratic sorting algorithms (selection, insertion) *[Fundamental computing algorithms]*

AED Algoritmos e Estruturas de Dados (2/2)
Object-oriented design *[Object-oriented programming]*
Encapsulation and information-hiding *[Object-oriented programming]*
Separation of behavior and implementation *[Object-oriented programming]*
Collection classes and iteration protocols *[Object-oriented programming]*
The conception of types as a set of values with together with a set of operations *[Declarations and types]*
Declaration models (binding, visibility, scope, and lifetime) *[Declarations and types]*
Overview of type-checking *[Declarations and types]*
Procedures, functions, and iterators as abstraction mechanisms *[Abstraction mechanisms]*
Parameterization mechanisms (reference vs. value) *[Abstraction mechanisms]*
Type parameters and parameterized types *[Abstraction mechanisms]*
O(N log N) sorting algorithms (Quicksort, heapsort, mergesort) *[Fundamental computing algorithms]*
Hash tables, including collision-avoidance strategies *[Fundamental computing algorithms]*
Binary search trees *[Fundamental computing algorithms]*
Asymptotic analysis of upper and average complexity bounds *[Basic algorithmic analysis]*

iv. Engenharia de Software [AMA]

iv.1 Introdução

A sub-área de Engenharia de Software na Secção de Informática abrange as disciplinas da área científica da LEIC com o mesmo nome, nomeadamente Engenharia de Software (ES), Laboratório Engenharia de Software (LES), Laboratório de Gestão de Projectos (LGP), Métodos Formais de Especificação (MFE) e Arquitecturas de Software (AS), e ainda a disciplina da LEEC designada por Engenharia de Software (ES).
As disciplinas desta sub-área incidem exclusivamente nos tópicos da área de conhecimento CC2001 designada por Software Engineering (SE), cobrindo alguns dos seus tópicos de forma aprofundada e os restantes apenas de forma superficial.
O conjunto de disciplinas da LEIC desta sub-área pretende transmitir conhecimentos e capacidades que permitam desempenhar qualquer um dos papeis típicos que um engenheiro de software pode assumir (analista, arquitecto, programador, gestor de produto, gestor de projecto) no desenvolvimento de produtos de software (código, documentação, manuais, etc) num ambiente condicionado em termos de recursos (custo, prazo, qualidade, fiabilidade, etc).
Os tópicos mais aprofundados são os directamente relacionados com o processo de construção de software, nomeadamente engenharia de requisitos, análise, desenho e construção de software. Os tópicos abordados de forma mais superficial, por questões de espaço curricular, são os relacionados com actividades que acompanham todo o processo de desenvolvimento, ao longo de todas as suas fases, tais como modelos de software, gestão de projectos, teste de software e qualidade de software.

iv.2 LEIC

iv.2.1 Conteúdo

ES Engenharia de Software (3/1)
Software e Engenharia de Software. Modelos de processos de desenvolvimento de software. Engenharia de requisitos. Desenho de software em UML. Teste de software. Evolução de Software.

LES Laboratório de Engenharia de Software (4/1)

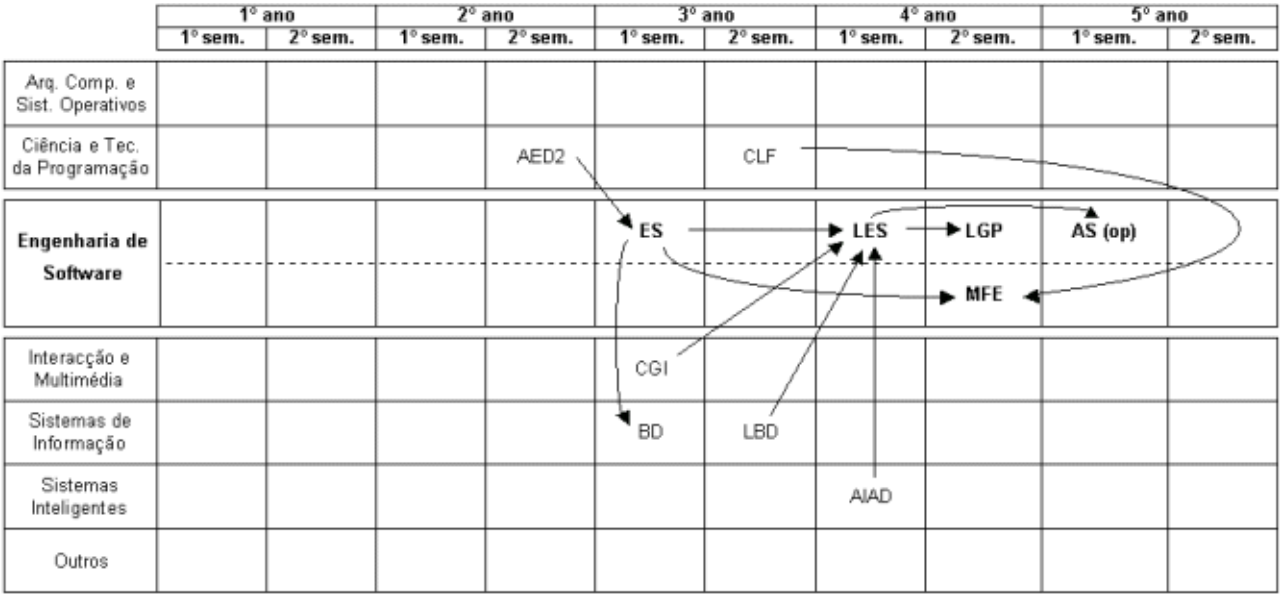
Desenvolvimento de um projecto de software (análise, desenho, construção, teste e documentação) integrando conhecimentos anteriores. Configuração de um processo de desenvolvimento de software. Ferramentas CASE. Utilização de API's complexas.

LGP Laboratório de Gestão de Projectos (4/2)
Desenvolvimento de um projecto de software cobrindo todo o ciclo de vida. Gestão de projectos: organização e colaboração. Planeamento de projectos. Condução e conclusão de projectos. Ferramentas.

MFE Métodos Formais de Especificação (4/2)
Especificação de software com recurso a métodos formais. Pré e pós-condições. Técnicas formais para validação, teste e verificação de software. Fiabilidade de software.

AS Arquitecturas de Software (5/1)
Desenho de Software. Arquitectura de software: nível macro (modelos de referência, estilos, padrões de arquitectura) e nível micro (padrões de desenho, componentes). Reutilização. Reengenharia. Desenvolvimento baseado em componentes.

iv.2.2 Análise de Dependências



Análise das Dependências para Engenharia de Software (LEIC)

iv.2.3 Comentários e Recomendações

Apesar de a primeira cadeira desta sub-área de Engenharia de Software apenas aparecer no 3º ano do plano curricular, alguns conhecimentos fundamentais desta área devem começar a ser transmitidos anteriormente, sobretudo em disciplinas da área de programação.
Em articulação com outras disciplinas, é recomendável explorar oportunidades para abordar e aplicar tópicos desta área de conhecimento em disciplinas anteriores nas quais a componente de desenvolvimento de software esteja presente, independentemente do seu domínio de aplicação, por forma a que os conhecimentos importantes da área possam ser devidamente consolidados e aprofundados até ao final do plano curricular. Exemplos de oportunidades destas são trabalhos de desenvolvimento de software propostos no âmbito de outras disciplinas, em que o domínio de aplicação é específico dessa disciplina, mas o desenvolvimento é proposto de forma a poder promover a compreensão e aplicação de um determinado tópico do corpo de conhecimentos de Engenharia de Software.

iv.2.4 Disciplinas

ES Engenharia de Software (3/1)
Software life-cycle and process models *[Software processes]*
Process assessment models *[Software processes]*
Software process metrics *[Software processes]*
Requirements elicitation *[Software requirements and specifications]*
Requirements analysis modeling techniques *[Software requirements and specifications]*
Functional and nonfunctional requirements *[Software requirements and specifications]*
Prototyping *[Software requirements and specifications]*
Basic concepts of formal specification techniques *[Software requirements and specifications]*
Object-oriented analysis and design *[Software design]*
Software architecture *[Software design]*
Design for reuse *[Software design]*
Testing fundamentals, including test plan creation and test case generation *[Software validation]*
Black-box and white-box testing techniques *[Software validation]*
Unit, integration, validation, and system testing *[Software validation]*
Object-oriented testing *[Software validation]*
Team management: Team processes, Team organization and decision-making, Roles and responsibilities in a software team, Role identification and assignment, Project tracking, Team problem resolution *[Software project management]*
Project scheduling *[Software project management]*
Software measurement and estimation techniques *[Software project management]*
Risk analysis *[Software project management]*
Software quality assurance *[Software project management]*
Software maintenance *[Software evolution]*
Characteristics of maintainable software *[Software evolution]*
Legacy systems *[Software evolution]*
Reengineering *[Software evolution]*
Software reuse *[Software evolution]*
Programming environments *[Software tools and environments]*
Requirements analysis and design modeling tools *[Software tools and environments]*
Testing tools *[Software tools and environments]*

LES Laboratório de Engenharia de Software (4/1)
Software life-cycle and process models *[Software processes]*
Process assessment models *[Software processes]*
Software process metrics *[Software processes]*
Object-oriented analysis and design *[Software design]*
Software architecture *[Software design]*
Design for reuse *[Software design]*
Component-level design *[Software design]*
Design patterns *[Software design]*
Programming environments *[Software tools and environments]*
Requirements analysis and design modeling tools *[Software tools and environments]*
Testing tools *[Software tools and environments]*
Tool integration mechanisms *[Software tools and environments]*
Introduction to component-based computing *[Using APIs]*
Class browsers and related tools *[Using APIs]*
Requirements elicitation *[Software requirements and specifications]*
Requirements analysis modeling techniques *[Software requirements and specifications]*
Prototyping *[Software requirements and specifications]*
Unit, integration, validation, and system testing *[Software validation]*
Object-oriented testing *[Software validation]*

LGP Laboratório de Gestão de Projectos (4/2)
Team management: Team processes, Team organization and decision-making, Roles and responsibilities in a software team, Role identification and assignment, Project tracking, Team

problem resolution *[Software project management]*
Project management tools *[Software project management]*
Project scheduling *[Software project management]*
Software measurement and estimation techniques *[Software project management]*
Risk analysis *[Software project management]*
Software quality assurance *[Software project management]*
Software configuration management *[Software project management]*
Software life-cycle and process models *[Software processes]*
Process assessment models *[Software processes]*
Software process metrics *[Software processes]*
Validation planning *[Software validation]*
Inspections *[Software validation]*
Unit, integration, validation, and system testing *[Software validation]*
Requirements elicitation *[Software requirements and specifications]*
Requirements analysis modeling techniques *[Software requirements and specifications]*
Functional and nonfunctional requirements *[Software requirements and specifications]*
Prototyping *[Software requirements and specifications]*
Software maintenance *[Software evolution]*
Characteristics of maintainable software *[Software evolution]*
Reengineering *[Software evolution]*
Software reuse *[Software evolution]*
Requirements analysis and design modeling tools *[Software tools and environments]*
Testing tools *[Software tools and environments]*
Configuration management tools *[Software tools and environments]*

MFE Métodos Formais de Especificação (4/2)
Formal methods concepts *[Formal methods]*
Formal specification languages *[Formal methods]*
Pre and post assertions *[Formal methods]*
Formal verification *[Formal methods]*
Executable and non-executable specifications *[Formal methods]*
Unit, integration, validation, and system testing *[Software validation]*
Basic concepts of formal specification techniques *[Software requirements and specifications]*
Software life-cycle and process models *[Software processes]*
Software reliability models *[Software reliability]*
Redundancy and fault tolerance *[Software reliability]*
Defect classification *[Software reliability]*
Probabilistic methods of analysis *[Software reliability]*

AS Arquitecturas de Software (5/1)
Fundamental design concepts and principles *[Software design]*
Design patterns *[Software design]*
Software architecture *[Software design]*
Component-level design *[Software design]*
Design for reuse *[Software design]*
Fundamentals: The definition and nature of components, Components and interfaces, Interfaces as contracts, The benefits of components *[Component-based computing]*
Basic techniques: Component design and assembly, Relationship with the client-server model and with patterns, Use of objects and object lifecycle services, Use of object brokers, Marshalling *[Component-based computing]*
Applications (including the use of mobile components) *[Component-based computing]*
Architecture of component-based systems *[Component-based computing]*
Component-oriented design *[Component-based computing]*
Event handling: detection, notification, and response *[Component-based computing]*
Middleware: The object-oriented paradigm within middleware, Object request brokers, Transaction processing monitors, Workflow systems, State-of-the-art tools *[Component-based computing]*

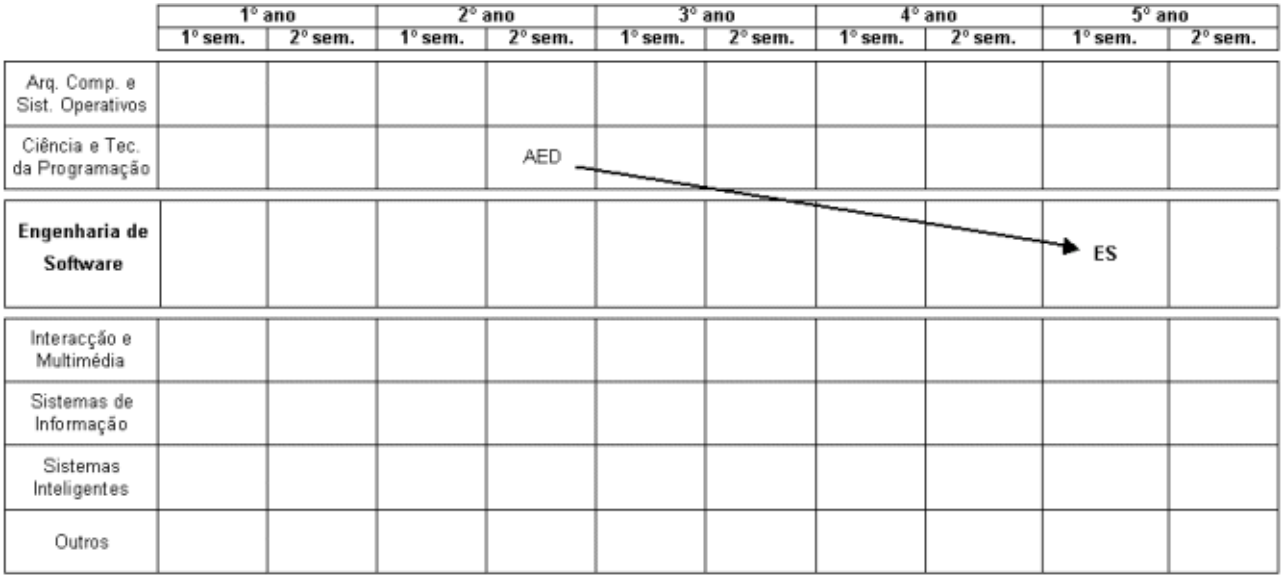
Introduction to component-based computing *[Using APIs]*
API programming *[Using APIs]*
Class browsers and related tools *[Using APIs]*
Programming by example *[Using APIs]*
Tool integration mechanisms *[Software tools and environments]*
Software reuse *[Software evolution]*
Reengineering *[Software evolution]*

iv.3 LEEC

iv.3.1 Conteúdo

ES2 Engenharia de Software (5/1)
Software e Engenharia de Software. Modelos de processos de desenvolvimento de software. Engenharia de requisitos. Desenho de software em UML. Teste de software. Evolução de Software.

iv.3.2 Análise de Dependências



Análise das Dependências para Engenharia de Software (LEEC)

iv.3.3 Comentários e Recomendações

Atendendo a que a única cadeira desta sub-área de Engenharia de Software apenas aparece no 5º ano do plano curricular, alguns conhecimentos fundamentais desta área devem começar a ser transmitidos anteriormente, sobretudo em disciplinas que envolvam programação. Em articulação com outras disciplinas anteriores, é essencial explorar oportunidades para abordar e aplicar tópicos desta área de conhecimento nas quais a componente de desenvolvimento de software esteja presente, independentemente do seu domínio de aplicação, por forma a que os conhecimentos importantes da área possam ser minimamente apreendidos. Exemplos de oportunidades destas são trabalhos de desenvolvimento de software propostos no âmbito de outras disciplinas, em que o domínio de aplicação é específico dessa disciplina, mas o desenvolvimento é proposto de forma a poder promover a compreensão e aplicação de um determinado tópico do corpo de conhecimentos de Engenharia de Software.

iv.3.4 Disciplinas

ES2 Engenharia de Software (5/1)
Software life-cycle and process models *[Software processes]*
Software process metrics *[Software processes]*
Requirements elicitation *[Software requirements and specifications]*
Requirements analysis modeling techniques *[Software requirements and specifications]*
Functional and nonfunctional requirements *[Software requirements and specifications]*
Prototyping *[Software requirements and specifications]*
Basic concepts of formal specification techniques *[Software requirements and specifications]*
Object-oriented analysis and design *[Software design]*
Software architecture *[Software design]*
Design for reuse *[Software design]*
Testing fundamentals, including test plan creation and test case generation *[Software validation]*
Black-box and white-box testing techniques *[Software validation]*
Unit, integration, validation, and system testing *[Software validation]*
Object-oriented testing *[Software validation]*
Team management: Team processes, Team organization and decision-making, Roles and responsibilities in a software team, Role identification and assignment, Project tracking, Team problem resolution *[Software project management]*
Project scheduling *[Software project management]*
Software measurement and estimation techniques *[Software project management]*
Risk analysis *[Software project management]*
Software quality assurance *[Software project management]*
Software maintenance *[Software evolution]*
Characteristics of maintainable software *[Software evolution]*
Legacy systems *[Software evolution]*
Reengineering *[Software evolution]*
Software reuse *[Software evolution]*
Programming environments *[Software tools and environments]*
Requirements analysis and design modeling tools *[Software tools and environments]*
Testing tools *[Software tools and environments]*

v. Interacção e Multimédia [AAS]

v.1 Introdução

A Sub-área "Interacção e Multimédia" integra a disciplina SG (Sistemas Gráficos) da LEEC e as disciplinas CGI (Computação Gráfica e Interfaces), LCp (Laboratório de Computação), IPC (Interacção Pessoa-Computador) e RV (Realidade Virtual), da LEIC.
No caso da LEIC, o conjunto definido de disciplinas pretende dotar os alunos com um conjunto de conhecimentos teóricos que abranjam os princípios gerais da Interacção, as bases da Computação Gráfica (com predomínio para os 3D) e uma perspectiva dos sistemas de Realidade Virtual. As vertentes mais práticas de demonstração dos princípios teóricos e dos algoritmos devem acompanhar as respectivas matérias das disciplinas mais teóricas. Numa perspectiva de integração e de multidisciplinaridade, a disciplina LCp engloba trabalhos práticos mais volumosos de Computação Gráfica e de outras disciplinas (de momento, Sistemas Operativos).

v.2 LEIC

v.2.1 Conteúdo

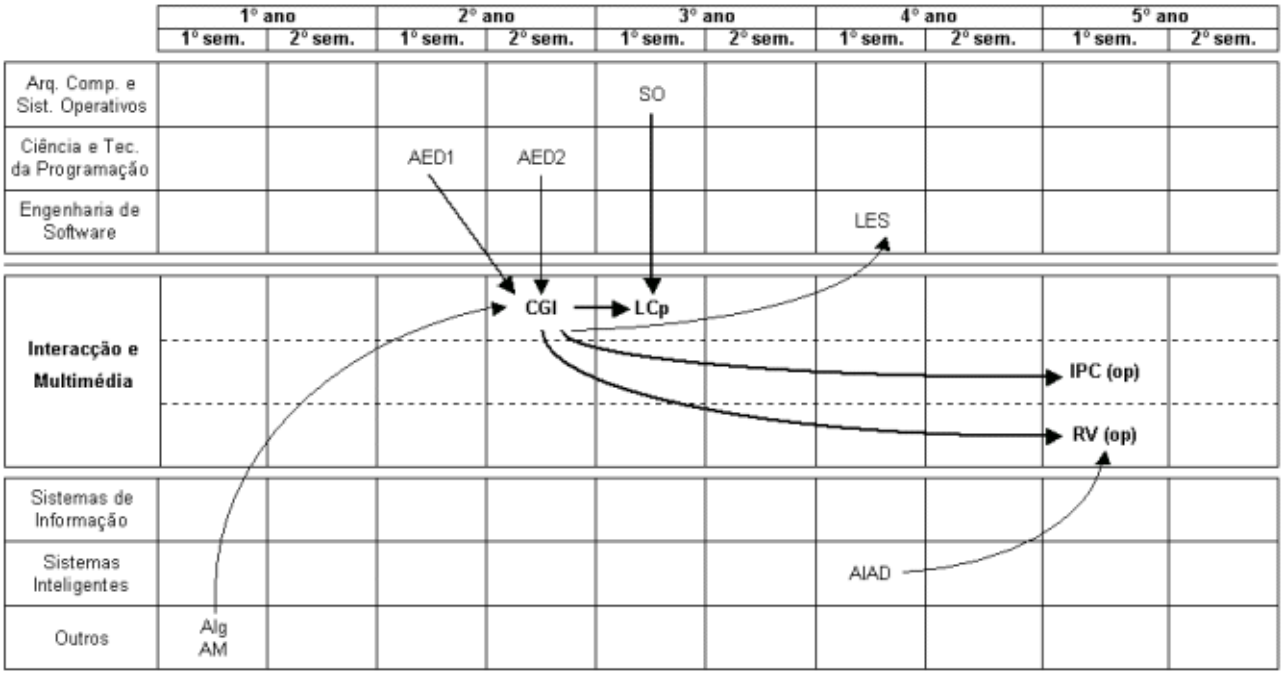
CGI Computação Gráfica e Interfaces (2/2)
Fundamentos de interacção pessoa-máquina; Técnicas fundamentais da Computação Gráfica; Modelação 3D, síntese de imagem e animação.

LCp Laboratório de Computação (3/1)
Desenvolvimento de projectos multidisciplinares.

IPC Interacção Pessoa-Computador (5/1)
Interacção pessoa-computador; Desenvolvimento e avaliação de software orientado ao utilizador; Desenho e programação de interfaces gráficas.

RV Realidade Virtual (5/1)
Introdução à realidade virtual; Fisiologia e percepção em ambientes virtuais; Dispositivos de Interacção em realidade virtual; Algoritmia de computação gráfico no contexto de realidade virtual; Aplicações.

v.2.2 Análise de Dependências



Análise das Dependências para Interacção e Multimédia (LEIC)

v.2.3 Comentários e Recomendações

Os trabalhos práticos que envolvem desenvolvimento de interfaces gráficas tendem a ser volumosos, sendo que grande parte deste volume se deve à gestão dos eventos associados com a interacção. Assim, é com alguma facilidade que os alunos se distraem com actividades que não são as mais importantes no contexto de um determinado trabalho. Recomenda-se que, quando adequado e no sentido de evitar situações como a anterior, sejam utilizados, no desenvolvimento dos trabalhos práticos de menor dimensão, sistemas de programação visual que automatizam grande parte do processamento de eventos. O fornecimento de exemplos no mesmo formato que o código a desenvolver pode também ser uma ajuda importante à rápida execução dos trabalhos e até mesmo à sua estruturação.

v.2.4 Disciplinas

CGI Computação Gráfica e Interfaces (2/2)
Hierarchy of graphics software *[Fundamental techniques in graphics]*
Simple color models (RGB, HSB, CMYK) *[Fundamental techniques in graphics]*
Homogeneous coordinates *[Fundamental techniques in graphics]*
Affine transformations (scaling, rotation, translation) *[Fundamental techniques in graphics]*
Viewing transformation *[Fundamental techniques in graphics]*

Clipping *[Fundamental techniques in graphics]*
Motivation: Why care about people? *[Foundations of human-computer interaction]*
Contexts for HCI (tools, web hypermedia, communication) *[Foundations of human-computer interaction]*
Human-centered development and evaluation *[Foundations of human-computer interaction]*
Accommodating human diversity *[Foundations of human-computer interaction]*
Principles of good design and good designers; engineering tradeoffs *[Foundations of human-computer interaction]*
Introduction to usability testing *[Foundations of human-computer interaction]*
Principles of graphical user interfaces (GUIs) *[Building a simple graphical user interface]*
GUI toolkits *[Building a simple graphical user interface]*
Line generation algorithms (Bresenham) *[Basic rendering]*
Scan conversion of 2D primitive, forward differencing *[Advanced techniques]*
Hidden surface removal methods *[Advanced techniques]*
Z-buffer and frame buffer, color channels (a channel for opacity) *[Advanced techniques]*
Image synthesis, sampling techniques, and anti-aliasing *[Basic rendering]*
Light-source and material properties *[Basic rendering]*
Ambient, diffuse, and specular reflections *[Basic rendering]*
Phong reflection model *[Basic rendering]*
Rendering of a polygonal surface; flat, Gouraud, and Phong shading *[Basic rendering]*
Texture mapping, bump texture, environment map *[Basic rendering]*
Introduction to ray tracing *[Basic rendering]*
Polygonal representation of 3D objects *[Geometric modeling]*
Implicit representation of curves and surfaces *[Geometric modeling]*
Parametric polynomial curves and surfaces *[Geometric modeling]*
Constructive Solid Geometry (CSG) representation *[Geometric modeling]*
Spatial subdivision techniques *[Geometric modeling]*
Procedural models *[Geometric modeling]*
Deformable models *[Geometric modeling]*
Transport equations *[Advanced rendering]*
Efficient approaches to global illumination *[Advanced rendering]*
Ray tracing algorithms *[Advanced rendering]*
Radiosity for global illumination computation, form factors *[Advanced rendering]*
Photon tracing *[Advanced rendering]*
Monte Carlo methods for global illumination *[Advanced rendering]*
Image-based rendering, panorama viewing, plenoptic function modeling *[Advanced rendering]*
Key-frame animation *[Computer animation]*
Camera animation *[Computer animation]*
Animation of articulated structures: inverse kinematics *[Computer animation]*
Deformation *[Computer animation]*
Procedural animation *[Computer animation]*

LCp Laboratório de Computação (3/1)
Image synthesis, sampling techniques, and anti-aliasing *[Basic rendering]*
Phong reflection model *[Basic rendering]*
Rendering of a polygonal surface; flat, Gouraud, and Phong shading *[Basic rendering]*
Texture mapping, bump texture, environment map *[Basic rendering]*
Functionality and usability: task analysis, interviews, surveys *[Human-centered software development]*
Prototyping techniques and tools: Paper storyboards, Inheritance and dynamic dispatch,
Prototyping languages and GUI builders *[Human-centered software development]*
Event management and user interaction *[Graphical user-interface programming]*
GUI builders and UI programming environments *[Graphical user-interface programming]*
Concepts of application program interfaces (APIs) *[Operating system principles]*
Processes and threads *[Scheduling and dispatch]*
The "mutual exclusion" problem and some solutions *[Concurrency]*
Models and mechanisms (semaphores, monitors, condition variables, rendezvous)

[Concurrency]
Producer-consumer problems and synchronization *[Concurrency]*
Memory-mapped files *[File systems]*
Directories: contents and structure *[File systems]*

IPC Interacção Pessoa-Computador (5/1)
Motivation: Why care about people? *[Foundations of human-computer interaction]*
Contexts for HCI (tools, web hypermedia, communication) *[Foundations of human-computer interaction]*
Human-centered development and evaluation *[Foundations of human-computer interaction]*
Accommodating human diversity *[Foundations of human-computer interaction]*
Principles of good design and good designers; engineering tradeoffs *[Foundations of human-computer interaction]*
Introduction to usability testing *[Foundations of human-computer interaction]*
Setting goals for evaluation *[Human-centered software evaluation]*
Evaluation without users: walkthroughs, KLM, guidelines, and standards *[Human-centered software evaluation]*
Evaluation with users: usability testing, interview, survey, experiment *[Human-centered software evaluation]*
Approaches, characteristics, and overview of process *[Human-centered software development]*
Functionality and usability: task analysis, interviews, surveys *[Human-centered software development]*
Specifying interaction and presentation *[Human-centered software development]*
Prototyping techniques and tools: Paper storyboards, Inheritance and dynamic dispatch,
Prototyping languages and GUI builders *[Human-centered software development]*
Choosing interaction styles and interaction techniques *[Graphical user-interface design]*
HCI aspects of common widgets *[Graphical user-interface design]*
HCI aspects of screen design: layout, color, fonts, labeling *[Graphical user-interface design]*
Handling human failure *[Graphical user-interface design]*
Beyond simple screen design: visualization, representation, metaphor *[Graphical user-interface design]*
Multi-modal interaction: graphics, sound, and haptics *[Graphical user-interface design]*
3D interaction and virtual reality *[Graphical user-interface design]*
UIMS, dialogue independence and levels of analysis, Seeheim model *[Graphical user-interface programming]*
Widget classes *[Graphical user-interface programming]*
Event management and user interaction *[Graphical user-interface programming]*
Geometry management *[Graphical user-interface programming]*
GUI builders and UI programming environments *[Graphical user-interface programming]*
Cross-platform design *[Graphical user-interface programming]*

RV Realidade Virtual (5/1)
Identifying virtual environments *[Introduction to Virtual Reality]*
Terminology of virtual reality *[Introduction to Virtual Reality]*
Synthesis of virtual environments *[Introduction to Virtual Reality]*
System architectures overview *[Introduction to Virtual Reality]*
Physiology of virtual perception *[Physiology and perception in virtual environments]*
Physiology of auditory perception *[Physiology and perception in virtual environments]*
Physiology of haptic and kinaesthetic perception *[Physiology and perception in virtual environments]*
Virtual presence *[Physiology and perception in virtual environments]*
Visual devices *[Interaction devices for virtual reality]*
Auditory devices *[Interaction devices for virtual reality]*
Haptic and kinaesthetic devices *[Interaction devices for virtual reality]*
Visibility computation *[Virtual reality]*
Collision detection *[Virtual reality]*

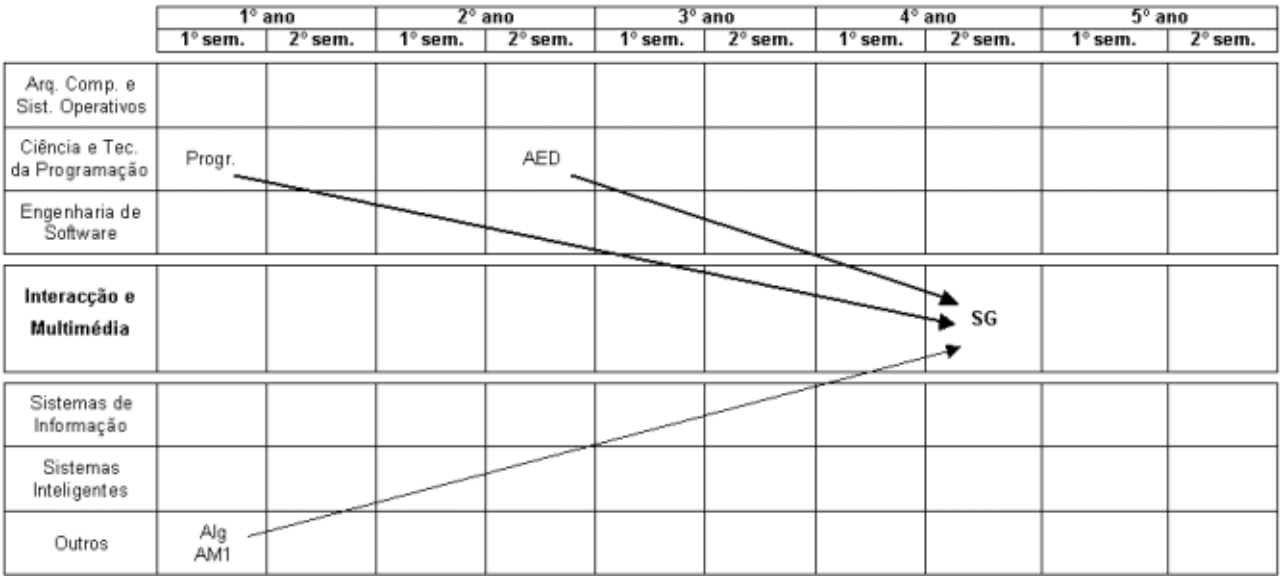
Time-critical rendering, multiple levels of details (LOD) *[Virtual reality]*
Image-base VR system *[Virtual reality]*
User interface issues *[Virtual reality]*
Applications in medicine, simulation, and training *[Virtual reality]*

v.3 LEEC

v.3.1 Conteúdo

SG Sistemas Gráficos (5/1)
Fundamentos de interacção pessoa-máquina; Técnicas fundamentais da Computação Gráfica; Modelação 3D, síntese de imagem e animação.

v.3.2 Análise de Dependências



Análise das Dependências para Interacção e Multimédia (LEEC)

v.3.3 Comentários e Recomendações

A disciplina SG é a única desta subárea que os alunos da LEEC encontram no seu plano de estudos. Assim, torna-se necessário que tomem conhecimento, ainda que introdutório, com os tópicos principais destas áreas.

v.3.4 Disciplinas

SG Sistemas Gráficos (5/1)
Hierarchy of graphics software *[Fundamental techniques in graphics]*
Simple color models (RGB, HSB, CMYK) *[Fundamental techniques in graphics]*
Homogeneous coordinates *[Fundamental techniques in graphics]*
Affine transformations (scaling, rotation, translation) *[Fundamental techniques in graphics]*
Viewing transformation *[Fundamental techniques in graphics]*
Clipping *[Fundamental techniques in graphics]*
Motivation: Why care about people? *[Foundations of human-computer interaction]*
Contexts for HCI (tools, web hypermedia, communication) *[Foundations of human-computer interaction]*
Human-centered development and evaluation *[Foundations of human-computer interaction]*
Accommodating human diversity *[Foundations of human-computer interaction]*
Principles of good design and good designers; engineering tradeoffs *[Foundations of human-*

computer interaction]
Introduction to usability testing *[Foundations of human-computer interaction]*
Principles of graphical user interfaces (GUIs) *[Building a simple graphical user interface]*
GUI toolkits *[Building a simple graphical user interface]*
Line generation algorithms (Bresenham) *[Basic rendering]*
Scan conversion of 2D primitive, forward differencing *[Advanced techniques]*
Hidden surface removal methods *[Advanced techniques]*
Z-buffer and frame buffer, color channels (a channel for opacity) *[Advanced techniques]*
Image synthesis, sampling techniques, and anti-aliasing *[Basic rendering]*
Light-source and material properties *[Basic rendering]*
Ambient, diffuse, and specular reflections *[Basic rendering]*
Phong reflection model *[Basic rendering]*
Rendering of a polygonal surface; flat, Gouraud, and Phong shading *[Basic rendering]*
Texture mapping, bump texture, environment map *[Basic rendering]*
Introduction to ray tracing *[Basic rendering]*
Polygonal representation of 3D objects *[Geometric modeling]*
Implicit representation of curves and surfaces *[Geometric modeling]*
Parametric polynomial curves and surfaces *[Geometric modeling]*
Constructive Solid Geometry (CSG) representation *[Geometric modeling]*
Spatial subdivision techniques *[Geometric modeling]*
Procedural models *[Geometric modeling]*
Deformable models *[Geometric modeling]*
Transport equations *[Advanced rendering]*
Efficient approaches to global illumination *[Advanced rendering]*
Ray tracing algorithms *[Advanced rendering]*
Radiosity for global illumination computation, form factors *[Advanced rendering]*
Photon tracing *[Advanced rendering]*
Monte Carlo methods for global illumination *[Advanced rendering]*
Image-based rendering, panorama viewing, plenoptic function modeling *[Advanced rendering]*
Key-frame animation *[Computer animation]*
Camera animation *[Computer animation]*
Animation of articulated structures: inverse kinematics *[Computer animation]*
Deformation *[Computer animation]*
Procedural animation *[Computer animation]*

vi. Sistemas de Informação [GTD]

vi.1 Introdução

A sub-área dos Sistemas de Informação corresponde essencialmente à área do conhecimento Information Management do CC2001, embora, pelo seu carácter integrador, cubra também outras áreas de que se destaca a Net-Centric Computing. Pela mesma razão, foi necessário acrescentar alguns domínios do conhecimento de forma a cobrir a perspectiva organizacional dos SI e a introdução ao uso dos computadores para não informáticos.

vi.2 LEIC

vi.2.1 Conteúdo

BD Bases de Dados (3/1)
Projecto de bases de dados. Álgebra relacional e SQL. Tecnologia de sistemas de informação e bases de dados.

LBD Laboratório de Bases de Dados (3/2)
Construção de repositórios de informação na Web. Projecto, implementação e documentação de um sistema de informação. Utilização de ferramentas CASE. Optimização de interrogações

. Modelo relacional-objecto.

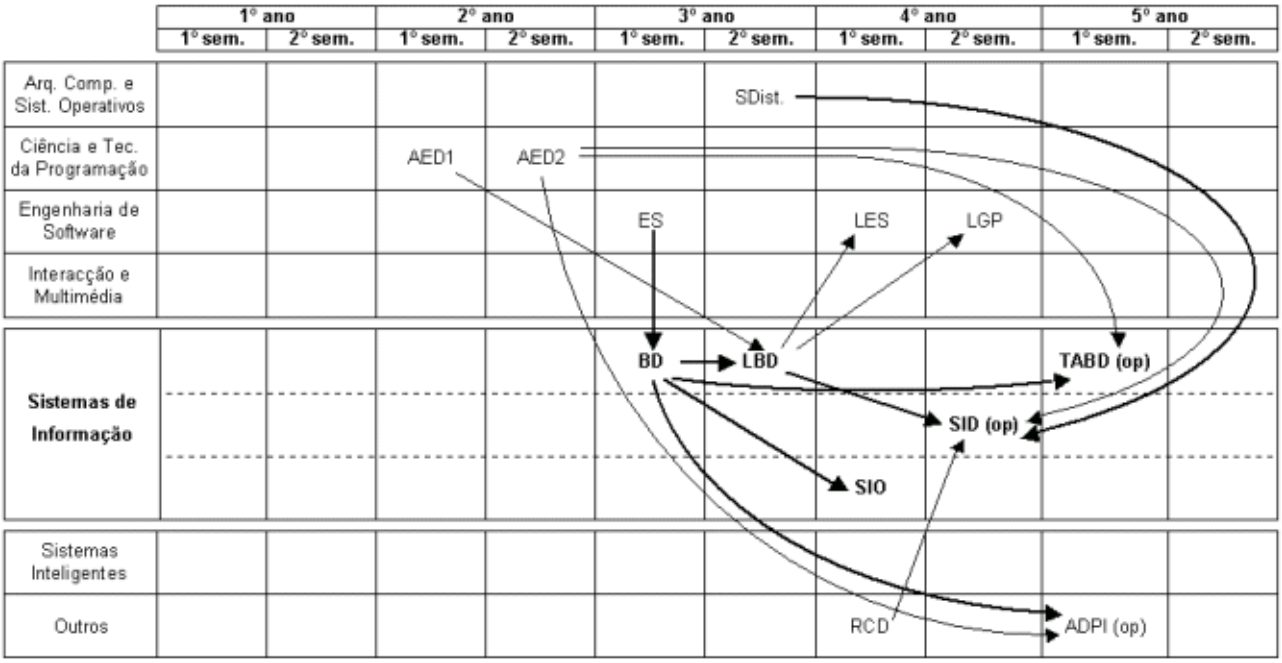
SIO Sistemas de Informação nas Organizações (4/1)
Sistemas de informação nas organizações. Os sistemas de informação e os processos de mudança. Avaliação de sistemas de informação. Implicações sociais das tecnologias de informação.

SID Sistemas de Informação Distribuídos (4/2)
A Web como um exemplo de aplicações cliente/servidor. Construção de aplicações na Web usando Java e XML. Acesso a bases de dados.

ADP Armazéns de Dados e Pesquisa de Informação (5/1)
Armazenamento e recuperação de informação: estruturas de dados; classificação; indexação; pesquisa. Armazenamento massivo de dados: limpeza de dados; prospecção de dados; visualização.

TABD Tópicos Avançados de Bases de Dados (5/1)
Estruturas físicas. Optimização de interrogações. Gestão de transacções, concorrência e recuperação. A norma ODMG2.0. Modelo relacional-objecto e SQL3. Bases de dados distribuídas.

vi.2.2 Análise de Dependências



Análise das Dependências para Sistemas de Informação (LEIC)

vi.2.3 Comentários e Recomendações

As disciplinas desta área constituem uma linha de carácter mais tecnológico que se inicia em Bases de Dados, continua em Laboratórios de Bases de Dados e termina em duas optativas, de Tópicos Avançados de Bases de Dados e de Armazéns de Dados e Pesquisa de Informação (esta não a cargo da Secção de Informática).
Uma segunda linha, progressivamente mais próxima da primeira, inclui as tecnologias de organização e acesso à informação pela Web (SI Distribuídos). Aliás, por razões de sequenciação, incluiu-se uma primeira abordagem às tecnologias da Web na disciplina de LBD, apesar de o assunto principal desta disciplina ser o desenvolvimento de SI através do uso de ferramentas apropriadas. [O número de tópicos nas duas vertentes da disciplina está muito desequilibrado, uma vez que a maior parte dos assuntos praticados em LBD foram

estudados em BD, mas privilegiou-se a cobertura dos tópicos do CC2001 em detrimento do seu equilíbrio face ao tempo que lhes é dedicado nas disciplinas.] De qualquer forma, algumas dessas ferramentas produzem precisamente aplicações na Web.
Existe uma terceira linha que se refere à perspectiva organizacional dos SI e de que a principal disciplina é Sistemas de Informação nas Organizações. Embora no plano de estudos da LEIC esta disciplina esteja classificada numa área científica de Gestão, a perspectiva que parece mais adequada para os alunos típicos da LEIC é precisamente a que parte dos conhecimentos tecnológicos já adquiridos nas disciplinas de BD e ES e os enquadra fundamentadamente numa perspectiva de Gestão, pelo que não se encontra afastada dos interesses da Secção de Informática.

Há um elevado número de dependências com disciplinas de áreas próximas, desde os requisitos de Programação, Sistemas Distribuídos e Redes até ao fornecimento de competências úteis para disciplinas de laboratórios. É de realçar uma parceria especial com Engenharia de Software, que decorre em simultâneo com BD, e em que se recomenda uma coordenação da evolução de ambas as disciplinas de forma a evitar sobreposições e a facilitar sinergias que poderão ser corporizadas em trabalhos práticos comuns.

vi.2.4 Disciplinas

BD Bases de Dados (3/1)
History and motivation for database systems *[Database systems]*
Components of database systems *[Database systems]*
DBMS functions *[Database systems]*
Database architecture and data independence *[Database systems]*
Use of a database query language *[Database systems]*
Data modeling *[Data modeling]*
Conceptual models (including entity-relationship and UML) *[Data modeling]*
Object-oriented model *[Data modeling]*
Relational data model *[Data modeling]*
Mapping conceptual schema to a relational schema *[Relational databases]*
Entity and referential integrity *[Relational databases]*
Relational algebra and relational calculus *[Relational databases]*
Overview of database languages *[Database query languages]*
SQL (data definition, query formulation, update sublanguage, constraints, integrity) *[Database query languages]*
Database design *[Relational database design]*
Functional dependency *[Relational database design]*
Normal forms (1NF, 2NF, 3NF, BCNF) *[Relational database design]*
Multivalued dependency (4NF) *[Relational database design]*
Join dependency (PJNF, 5NF) *[Relational database design]*
Representation theory *[Relational database design]*

LBD Laboratório de Bases de Dados (3/2)
Hypertext models (early history, web, Dexter, Amsterdam, HyTime) *[Hypertext and hypermedia]*
Link services, engines, and (distributed) hypertext architectures *[Hypertext and hypermedia]*
Nodes, composites, and anchors *[Hypertext and hypermedia]*
Dimensions, units, locations, spans *[Hypertext and hypermedia]*
Browsing, navigation, views, zooming *[Hypertext and hypermedia]*
Automatic link generation *[Hypertext and hypermedia]*
Presentation, transformations, synchronization *[Hypertext and hypermedia]*
Authoring, reading, and annotation *[Hypertext and hypermedia]*
Protocols and systems (including web, HTTP) *[Hypertext and hypermedia]*
QBE and 4th-generation environments *[Database query languages]*
Embedding non-procedural queries in a procedural language *[Database query languages]*

SIO Sistemas de Informação nas Organizações (4/1)
Basic types of applications in the organisation *[Information systems in the organisation]*
Information systems from a functional perspective *[Information systems in the organisation]*
Functional integration, business processes and networked organisations *[Information systems in the organisation]*
The relationship between the organisation, management and information systems *[Information systems in the organisation]*
Strategic information systems *[Information systems in the organisation]*
Business strategy and value chain model *[Information systems in the organisation]*
Information systems and organisational change *[Redesign the organisation with information systems]*
Information systems strategy and planning *[Redesign the organisation with information systems]*
Business process (re)engineering and continuous improvement *[Redesign the organisation with information systems]*
Approaches to socio-technical design *[Redesign the organisation with information systems]*
Business value of information systems *[Assessment of information systems in the organisation]*
Change management and critical success factors *[Assessment of information systems in the organisation]*
Management of the implementation *[Assessment of information systems in the organisation]*
Introduction to the social implications of computing *[Social context of computing]*
Social implications of networked communication *[Social context of computing]*
Understanding the social context of design *[Methods and tools of analysis]*
Identifying assumptions and values *[Methods and tools of analysis]*

SID Sistemas de Informação Distribuídos (4/2)
Background and history of networking and the Internet *[Introduction to net-centric computing]*
Network architectures *[Introduction to net-centric computing]*
The range of specializations within net-centric computing: Networks and protocols, Networked multimedia systems, Distributed computing, Mobile and wireless computing. *[Introduction to net-centric computing]*
Web technologies: Server-side programs, Common gateway interface (CGI) programs, Client-side scripts, The applet concept *[The web as an example of client-server computing]*
Characteristics of web servers: Handling permissions, File management, Capabilities of common server architectures. *[The web as an example of client-server computing]*
Role of client computers *[The web as an example of client-server computing]*
Nature of the client-server relationship *[The web as an example of client-server computing]*
Web protocols *[The web as an example of client-server computing]*
Support tools for web site creation and web management *[The web as an example of client-server computing]*
Developing Internet information servers *[The web as an example of client-server computing]*
Publishing information and applications *[The web as an example of client-server computing]*
Protocols at the application layer *[Building web applications]*
Principles of web engineering *[Building web applications]*
Database-driven web sites *[Building web applications]*
Remote procedure calls (RPC) *[Building web applications]*
Lightweight distributed objects *[Building web applications]*
The role of middleware *[Building web applications]*
Support tools *[Building web applications]*
Security issues in distributed object systems *[Building web applications]*
Enterprise-wide web-based applications *[Building web applications]*

ADP Armazéns de Dados e Pesquisa de Informação (5/1)
Characters, strings, coding, text *[Information storage and retrieval]*
Documents, electronic publishing, markup, and markup languages *[Information storage and retrieval]*

Tries, inverted files, PAT trees, signature files, indexing *[Information storage and retrieval]*
Morphological analysis, stemming, phrases, stop lists *[Information storage and retrieval]*
Term frequency distributions, uncertainty, fuzziness, weighting *[Information storage and retrieval]*
Vector space, probabilistic, logical, and advanced models *[Information storage and retrieval]*
Information needs, relevance, evaluation, effectiveness *[Information storage and retrieval]*
Thesauri, ontologies, classification and categorization, metadata *[Information storage and retrieval]*
Bibliographic information, bibliometrics, citations *[Information storage and retrieval]*
Routing and (community) filtering *[Information storage and retrieval]*
Search and search strategy, information seeking behavior, user modeling, feedback *[Information storage and retrieval]*
Information summarization and visualization *[Information storage and retrieval]*
Integration of citation, keyword, classification scheme, and other terms *[Information storage and retrieval]*
Protocols and systems (including Z39.50, OPACs, WWW engines, research systems) *[Information storage and retrieval]*
Data warehouse construction *[Data mining]*
The usefulness of data mining *[Data mining]*
Associative and sequential patterns *[Data mining]*
Data clustering *[Data mining]*
Market basket analysis *[Data mining]*
Data cleaning *[Data mining]*
Data visualization *[Data mining]*

TABD Tópicos Avançados de Bases de Dados (5/1)
Query optimization *[Database query languages]*
Introduction to Object Query Language *[Database query languages]*
Transactions *[Transaction processing]*
Failure and recovery *[Transaction processing]*
Concurrency control *[Transaction processing]*
Distributed data storage *[Distributed databases]*
Distributed query processing *[Distributed databases]*
Distributed transaction model *[Distributed databases]*
Concurrency control *[Distributed databases]*
Homogeneous and heterogeneous solutions *[Distributed databases]*
Client-server *[Distributed databases]*
Storage and file structure *[Physical database design]*
Indexed files *[Physical database design]*
Hashed files *[Physical database design]*
Signature files *[Physical database design]*
B-trees *[Physical database design]*
Files with dense index *[Physical database design]*
Files with variable length records *[Physical database design]*
Database efficiency and tuning *[Physical database design]*

vi.3 LEEC

vi.3.1 Conteúdo

FSI Fundamentos de Sistemas de Informação (4/1)
Sistemas operativos. Concorrência. Escalonamento. Gestão de memória. Sistemas de ficheiros. Bases de dados. Modelação de dados. Bases de dados relacionais. Linguagens de interrogação.

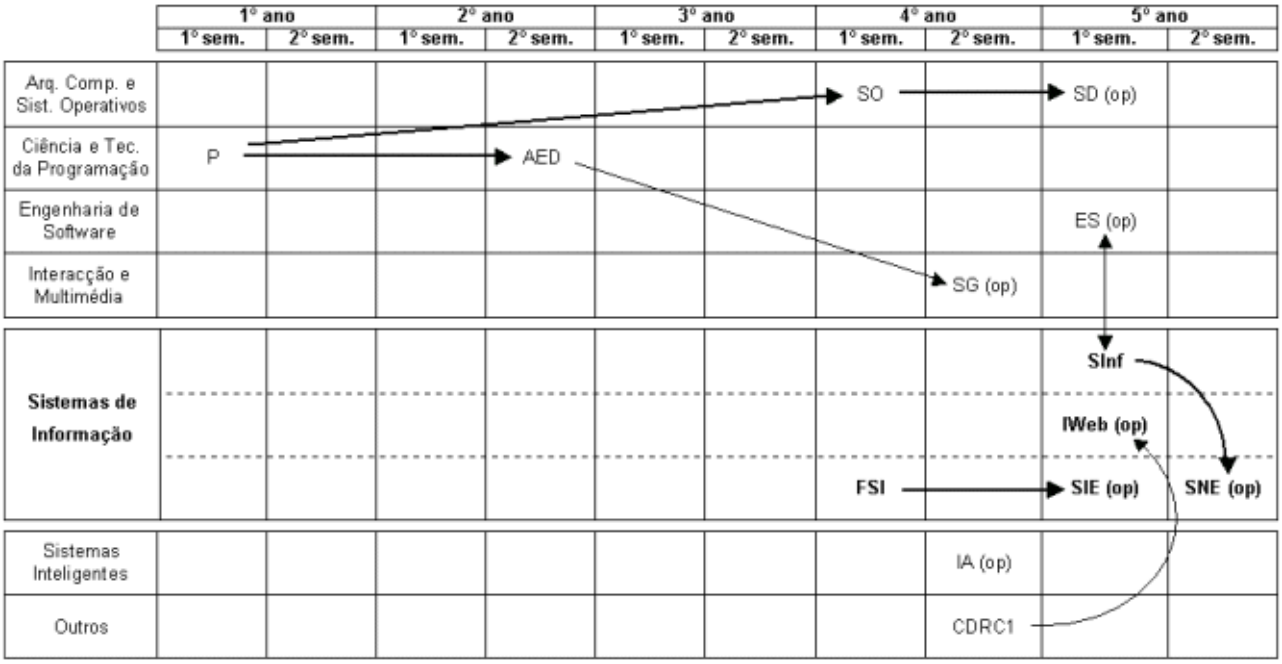
IW Informação na Web (5/1)
Construção de repositórios de informação na Web. Ferramentas e técnicas de extracção e

análise automática de informação recolhida na Web.

SI Sistemas de Informação (5/1)
Sistema de informação. Modelo entidade-associação. Modelo relacional. Projecto de bases de dados relacionais. SQL. Desenvolvimento de aplicações cliente-servidor. Privacidade e integridade dos dados. Transacções. Bases de dados relacional-objecto.

SIE Sistemas de Informação Empresarial (5/1)
Tecnologias e arquitecturas para Sistemas de Informação Empresarial. Sistemas de Informação Empresarial. Projecto da empresa integrada.

vi.3.2 Análise de Dependências



Análise das Dependências para Sistemas de Informação (LEEC)

vi.3.3 Comentários e Recomendações

Tal como referido na Introdução, incluem-se neste diagrama todas as disciplinas de Informática da LEEC, para facilidade de referência.
Não há propriamente linhas de disciplinas na sub-área de SI na LEEC, mas apenas disciplinas mais ou menos isoladas. É possível distinguir entre os três ramos do curso.
Os alunos de Energia apenas são sujeitos às duas disciplinas de Programação.
Os alunos de APEL, têm ainda uma sequência de duas disciplinas, FSI (obrigatória) cobrindo os sistemas operativos e as bases de dados e SIE (optativa), sobre os aspectos organizacionais e de acesso à informação através da Web.
O ramo de Telecomunicações tem uma cobertura maior de disciplinas de Informática. Existem 4 disciplinas obrigatórias, 2 de programação, SO e Sinf, para além de 5 disciplinas optativas, sem contar com SM: SDist, SG, ES, IA e IWeb.

vi.3.4 Disciplinas

FSI Fundamentos de Sistemas de Informação (4/1)
Role and purpose of the operating system *[Overview of operating systems]*
History of operating system development *[Overview of operating systems]*
Functionality of a typical operating system *[Overview of operating systems]*
Mechanisms to support client-server models, hand-held devices *[Overview of operating*

systems]
Design issues (efficiency, robustness, flexibility, portability, security, compatibility) *[Overview of operating systems]*
Influences of security, networking, multimedia, windows *[Overview of operating systems]*
Structuring methods (monolithic, layered, modular, micro-kernel models) *[Operating system principles]*
Abstractions, processes, and resources *[Operating system principles]*
Concepts of application program interfaces (APIs) *[Operating system principles]*
Application needs and the evolution of hardware/software techniques *[Operating system principles]*
Device organization *[Operating system principles]*
Interrupts: methods and implementations *[Operating system principles]*
Concept of user/system state and protection, transition to kernel mode *[Operating system principles]*
States and state diagrams *[Concurrency]*
Structures (ready list, process control blocks, and so forth) *[Concurrency]*
Dispatching and context switching *[Concurrency]*
The role of interrupts *[Concurrency]*
Concurrent execution: advantages and disadvantages *[Concurrency]*
The "mutual exclusion" problem and some solutions *[Concurrency]*
Deadlock: causes, conditions, prevention *[Concurrency]*
Models and mechanisms (semaphores, monitors, condition variables, rendezvous) *[Concurrency]*
Producer-consumer problems and synchronization *[Concurrency]*
Multiprocessor issues (spin-locks, reentrancy) *[Concurrency]*
Preemptive and nonpreemptive scheduling *[Scheduling and dispatch]*
Schedulers and policies *[Scheduling and dispatch]*
Processes and threads *[Scheduling and dispatch]*
Deadlines and real-time issues *[Scheduling and dispatch]*
Review of physical memory and memory management hardware *[Memory management]*
Overlays, swapping, and partitions *[Memory management]*
Paging and segmentation *[Memory management]*
Placement and replacement policies *[Memory management]*
Working sets and thrashing *[Memory management]*
Caching *[Memory management]*
Files: data, metadata, operations, organization, buffering, sequential, nonsequential *[File systems]*
Directories: contents and structure *[File systems]*
File systems: partitioning, mount/unmount, virtual file systems *[File systems]*
Standard implementation techniques *[File systems]*
Memory-mapped files *[File systems]*
Special-purpose file systems *[File systems]*
Naming, searching, access, backups *[File systems]*
Data modeling *[Data modeling]*
Conceptual models (including entity-relationship and UML) *[Data modeling]*
Object-oriented model *[Data modeling]*
Relational data model *[Data modeling]*
Mapping conceptual schema to a relational schema *[Relational databases]*
Entity and referential integrity *[Relational databases]*
Relational algebra and relational calculus *[Relational databases]*
Overview of database languages *[Database query languages]*
SQL (data definition, query formulation, update sublanguage, constraints, integrity) *[Database query languages]*
Query optimization *[Database query languages]*
Embedding non-procedural queries in a procedural language *[Database query languages]*

IW Informação na Web (5/1)

Hypertext models (early history, web, Dexter, Amsterdam, HyTime) *[Hypertext and hypermedia]*
Link services, engines, and (distributed) hypertext architectures *[Hypertext and hypermedia]*
Nodes, composites, and anchors *[Hypertext and hypermedia]*
Dimensions, units, locations, spans *[Hypertext and hypermedia]*
Browsing, navigation, views, zooming *[Hypertext and hypermedia]*
Automatic link generation *[Hypertext and hypermedia]*
Presentation, transformations, synchronization *[Hypertext and hypermedia]*
Authoring, reading, and annotation *[Hypertext and hypermedia]*
Protocols and systems (including web, HTTP) *[Hypertext and hypermedia]*
Web technologies: Server-side programs, Common gateway interface (CGI) programs, Client-side scripts, The applet concept *[The web as an example of client-server computing]*
Characteristics of web servers: Handling permissions, File management, Capabilities of common server architectures. *[The web as an example of client-server computing]*
Role of client computers *[The web as an example of client-server computing]*
Nature of the client-server relationship *[The web as an example of client-server computing]*
Web protocols *[The web as an example of client-server computing]*
Support tools for web site creation and web management *[The web as an example of client-server computing]*
Developing Internet information servers *[The web as an example of client-server computing]*
Publishing information and applications *[The web as an example of client-server computing]*
Protocols at the application layer *[Building web applications]*
Principles of web engineering *[Building web applications]*
Database-driven web sites *[Building web applications]*
Remote procedure calls (RPC) *[Building web applications]*
The role of middleware *[Building web applications]*
Enterprise-wide web-based applications *[Building web applications]*
Documents, electronic publishing, markup, and markup languages *[Information storage and retrieval]*

SI Sistemas de Informação (5/1)
History and motivation for information systems *[Information models and systems]*
Information storage and retrieval (IS&R) *[Information models and systems]*
Information management applications *[Information models and systems]*
Basic types of applications in the organisation *[Information systems in the organisation]*
Information systems from a functional perspective *[Information systems in the organisation]*
The relationship between the organisation, management and information systems *[Information systems in the organisation]*
History and motivation for database systems *[Database systems]*
Components of database systems *[Database systems]*
DBMS functions *[Database systems]*
Database architecture and data independence *[Database systems]*
Use of a database query language *[Database systems]*
Data modeling *[Data modeling]*
Conceptual models (including entity-relationship and UML) *[Data modeling]*
Object-oriented model *[Data modeling]*
Relational data model *[Data modeling]*
Mapping conceptual schema to a relational schema *[Relational databases]*
Entity and referential integrity *[Relational databases]*
Relational algebra and relational calculus *[Relational databases]*
SQL (data definition, query formulation, update sublanguage, constraints, integrity) *[Database query languages]*
QBE and 4th-generation environments *[Database query languages]*
Embedding non-procedural queries in a procedural language *[Database query languages]*
Database design *[Relational database design]*
Functional dependency *[Relational database design]*
Normal forms (1NF, 2NF, 3NF, BCNF) *[Relational database design]*

Transactions *[Transaction processing]*
Failure and recovery *[Transaction processing]*
Concurrency control *[Transaction processing]*
Information privacy, integrity, security, and preservation *[Information models and systems]*

SIE Sistemas de Informação Empresarial (5/1)
Basic types of applications in the organisation *[Information systems in the organisation]*
Information systems from a functional perspective *[Information systems in the organisation]*
Functional integration, business processes and networked organisations *[Information systems in the organisation]*
Enterprise modelling methods *[Enterprise Modelling and Integration]*
Business modelling using UML *[Enterprise Modelling and Integration]*
Concepts of business processes integration *[Enterprise Modelling and Integration]*
Introduction to enterprise resource planning and supply chain management systems *[Enterprise Modelling and Integration]*
Web technologies: Server-side programs, Common gateway interface (CGI) programs, Client-side scripts, The applet concept *[The web as an example of client-server computing]*
Characteristics of web servers: Handling permissions, File management, Capabilities of common server architectures. *[The web as an example of client-server computing]*
Role of client computers *[The web as an example of client-server computing]*
Nature of the client-server relationship *[The web as an example of client-server computing]*
Web protocols *[The web as an example of client-server computing]*
Support tools for web site creation and web management *[The web as an example of client-server computing]*
Developing Internet information servers *[The web as an example of client-server computing]*
Publishing information and applications *[The web as an example of client-server computing]*
Protocols at the application layer *[Building web applications]*
Principles of web engineering *[Building web applications]*
Database-driven web sites *[Building web applications]*
Remote procedure calls (RPC) *[Building web applications]*
Lightweight distributed objects *[Building web applications]*
The role of middleware *[Building web applications]*
Support tools *[Building web applications]*
Security issues in distributed object systems *[Building web applications]*
Enterprise-wide web-based applications *[Building web applications]*

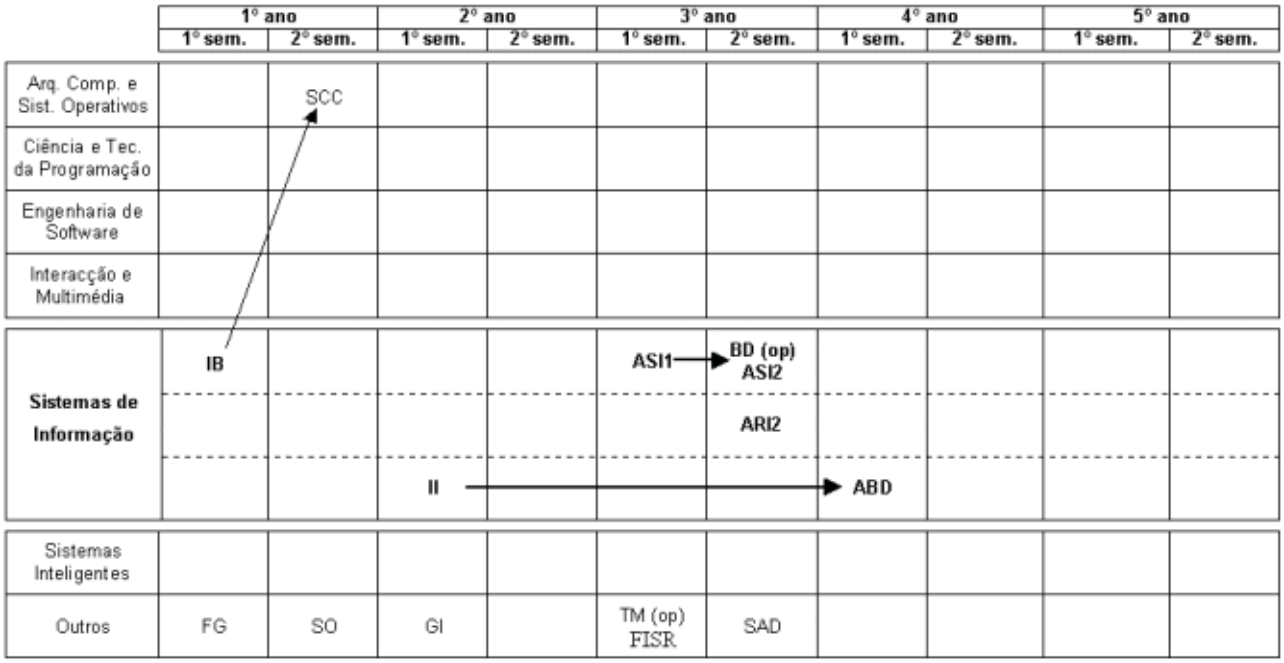
vi.4 LCI

vi.4.1 Conteúdo

IB Informática Básica (1/1)
Introdução aos computadores. Arquitectura. Elementos constituintes do computador actual. Ferramentas de escritório electrónico. Tratamento de gráficos e imagens.

II Informação na Internet (2/1)
A Internet: breve história. A World-Wide Web. Criação de documentos Web: Hypertext Markup Language (HTML); folhas de estilos. Criação de conteúdos dinâmicos; acesso a bases de dados. Novas linguagem de anotação: XML e XSL. Design. O futuro da Web.

vi.4.2 Análise de Dependências



Análise das Dependências para Sistemas de Informação (LCI)

vi.4.3 Comentários e Recomendações

Tal como referido na Introdução, incluem-se neste diagrama todas as disciplinas da responsabilidade da FEUP na LCI, com intuitos de apresentação, sendo que as que cabem ao DEEC são na sua maioria da SecInf. No entanto, só se detalham os programas das disciplinas já em funcionamento (em 2002/2003 funcionarão o 1º e 2º anos).
A perspectiva de Informática Básica é claramente distinta da generalidade das outras disciplinas, uma vez que se destina a dar a introdução à utilização de computadores a alunos que não são de Informática. Por essa razão, criou-se uma unidade de conhecimento específica.

vi.4.4 Disciplinas

IB Informática Básica (1/1)
Overview and history of computer architecture *[Digital logic and digital systems]*
Bits, bytes, and words *[Machine level representation of data]*
Numeric data representation and number bases *[Machine level representation of data]*
Fixed- and floating-point systems *[Machine level representation of data]*
Signed and twos-complement representations *[Machine level representation of data]*
Representation of nonnumeric data (character codes, graphical data) *[Machine level representation of data]*
Representation of records and arrays *[Machine level representation of data]*
Basic organization of the von Neumann machine *[Assembly level machine organization]*
Storage systems and their technology *[Memory system organization and architecture]*
Word processing *[Applications and Tools]*
Spreadsheets *[Applications and Tools]*
Presentation tools *[Applications and Tools]*
Image processing tools *[Applications and Tools]*
Personal databases *[Applications and Tools]*

II Informação na Internet (2/1)
Hypertext models (early history, web, Dexter, Amsterdam, HyTime) *[Hypertext and hypermedia]*
Link services, engines, and (distributed) hypertext architectures *[Hypertext and hypermedia]*

Nodes, composites, and anchors *[Hypertext and hypermedia]*
Dimensions, units, locations, spans *[Hypertext and hypermedia]*
Browsing, navigation, views, zooming *[Hypertext and hypermedia]*
Automatic link generation *[Hypertext and hypermedia]*
Presentation, transformations, synchronization *[Hypertext and hypermedia]*
Authoring, reading, and annotation *[Hypertext and hypermedia]*
Protocols and systems (including web, HTTP) *[Hypertext and hypermedia]*
Web technologies: Server-side programs, Common gateway interface (CGI) programs, Client-side scripts, The applet concept *[The web as an example of client-server computing]*
Characteristics of web servers: Handling permissions, File management, Capabilities of common server architectures. *[The web as an example of client-server computing]*
Role of client computers *[The web as an example of client-server computing]*
Nature of the client-server relationship *[The web as an example of client-server computing]*
Web protocols *[The web as an example of client-server computing]*
Support tools for web site creation and web management *[The web as an example of client-server computing]*
Developing Internet information servers *[The web as an example of client-server computing]*
Publishing information and applications *[The web as an example of client-server computing]*
Protocols at the application layer *[Building web applications]*
Principles of web engineering *[Building web applications]*
Database-driven web sites *[Building web applications]*
Remote procedure calls (RPC) *[Building web applications]*
The role of middleware *[Building web applications]*
Enterprise-wide web-based applications *[Building web applications]*
Documents, electronic publishing, markup, and markup languages *[Information storage and retrieval]*

vii. Sistemas Inteligentes [APR]

vii.1 Introdução

Na secção de Informática e no presente ano lectivo (2002/2003), a sub-área de Sistemas Inteligentes abrange disciplinas da área científica da LEIC com o mesmo nome: Inteligência Artificial (IA), Agentes e Inteligência Artificial Distribuída (AIAD), Complementos de Inteligência Artificial (CIA), Robótica (Rob) e Linguagem Natural e Tradução Automática (LNTA).
Na Licenciatura em Engenharia Electrotécnica e de Computadores (LEEC), está abrangida uma disciplina da área de Informática, Inteligência Artificial (IA) no ramo TEC.
Nos quadros seguintes resumem-se algumas das características destas disciplinas.

Licenciatura em Engenharia Informática e Computação (LEIC):

Área de Inteligência Artificial:
Inteligência Artificial (3º ano - 2º semestre) (3T + 1TP)
Agentes e Inteligência Artificial Distribuída (4º ano - 1º semestre) (2T + 2TP)
Robótica (op) (4º ano - 2º semestre) (3T + 1TP)
Complementos de Inteligência Artificial (op) (5º ano - 1º semestre) (3T + 1TP)
Linguagem Natural e Tradução Automática (op) (5º ano - 1º semestre) (3T + 1TP)

Licenciatura em Engenharia Electrotécnica e de Computadores

Área de Informática: **(Ramo de Sistemas de Telecomunicações, Electrónica e de Computadores)**
Inteligência Artificial (op) (4º ano - 2º semestre) (3T + 1TP)

As disciplinas da área de Sistemas Inteligentes (SI) abordam novas metodologias de representação do conhecimento e resolução de problemas baseadas em sistemas inteligentes. Na LEIC existem duas disciplinas de carácter obrigatório, que introduzem os conceitos fundamentais, incluindo resolução heurística de problemas (IA), aprendizagem (IA e AIAD) e teoria de agentes e sistemas multi-agente (AIAD). As restantes três disciplinas são optativas e abordam a área da robótica inteligente incluindo planeamento automático (Rob), o tratamento da linguagem natural (LNTA) e tópicos mais avançados nos assuntos já descritos em IA (CIA). Na LEEC existe uma única disciplina (IA) que é idêntica à do mesmo nome existente na LEIC.
As disciplinas desta área aparecem na última metade da licenciatura, após os alunos terem sido confrontados com outros paradigmas de IA como aquisição de conhecimento e resolução de problemas, e tendo já sido expostos ao paradigma lógico da programação (no caso da LEIC).

vii.2 LEIC

vii.2.1 Conteúdo

IA Inteligência Artificial (3/2)
Métodos de resolução de problemas. Representação de conhecimento. Métodos de raciocínio sobre conhecimento. Introdução ao processamento da linguagem natural. Aprendizagem simbólica e redes neuronais.

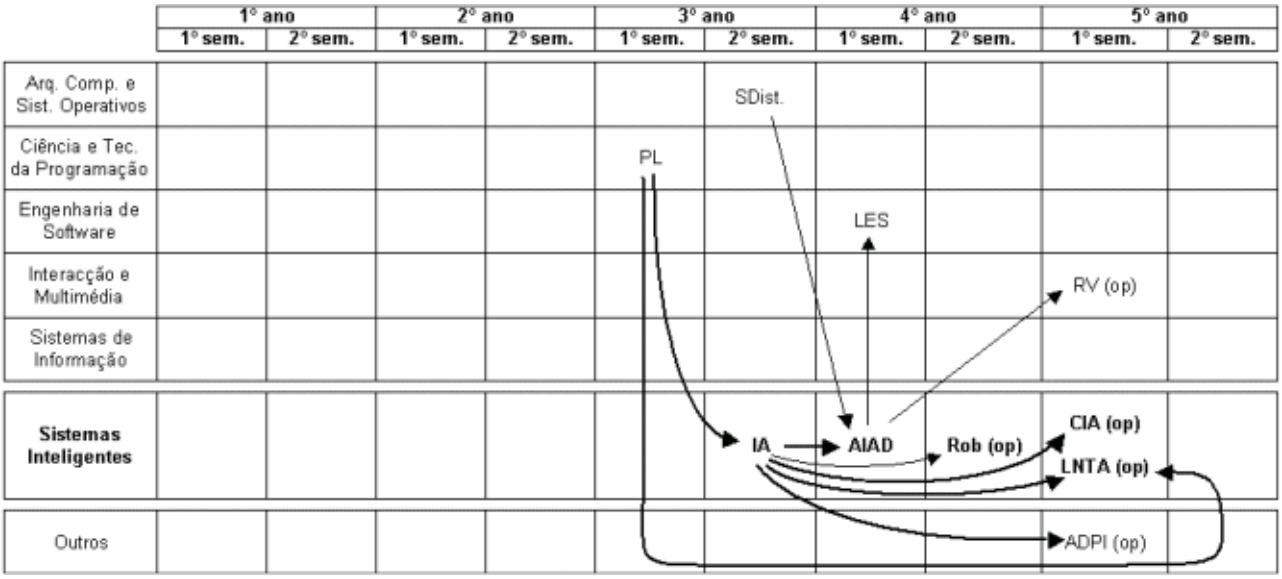
AIAD Agentes e IA Distribuída (4/1)
Métodos de resolução distribuída e cooperativa de problemas. Agentes e Sistemas Multi-agente. Representação de conhecimento em agentes. Protocolos de interacção e Ontologias. Agentes Adaptativos. Ferramentas de software para agentes distribuídos na Web.

Rob Robótica (4/2)
Métodos de planeamento inteligente. Geração automática de planos. Aprendizagem de planos. Arquitecturas de controlo: reactivas e deliberativas. Percepção, controlo e navegação em robótica. Linguagens robóticas.

CIA Complementos de Inteligência Artificial (5/1)
Resolução de problemas baseados em restrições. Métodos avançados de representação do conhecimento: Raciocínio não-monótono e por omissão. Sist. baseados em Conhecimento. Aprendizagem: Programação Lógica Indutiva e tópicos avançados de Redes Neuronais.

LNTA Linguagem Natural e Tradução Automática (5/1)
Representação de estruturas linguísticas. Gramáticas. Semântica. Sistemas de tradução automática. Técnicas probabilísticas na tradução.

vii.2.2 Análise de Dependências



Análise das Dependências para Sistemas Inteligentes (LEIC)

vii.2.3 Comentários e Recomendações

Recomenda-se que a disciplina de Laboratório de Engenharia de Software (LES) incentive o desenvolvimento de projectos de software também na área de Sistemas Inteligentes.
Para o desenvolvimento de trabalhos nas disciplinasda sub-área de Sistemas Inteligentes, nomeadamente AIAD e CIA, será útil que os alunos possuam conhecimentos de programação na Web.
a Aprendizagem das disciplinas de IA e IAD é concebida no sentido de obtenção de modelos lógicos de raciocínio e representação o mais generalizáveis possível. Daí que podem ser vitais como disciplinas também formativas.
É importante existir uma coordenação cuidada e conjunta dos conteúdos da disciplina de Programação em Lógica (PL) e das disciplinas da sub-secção de Sistemas Inteligentes.

vii.2.4 Disciplinas

IA Inteligência Artificial (3/2)
History of artificial intelligence *[Fundamental issues in intelligent systems]*
Philosophical questions *[Fundamental issues in intelligent systems]*
Fundamental definitions: Optimal vs. human-like reasoning, Optimal vs. human-like behavior *[Fundamental issues in intelligent systems]*
Philosophical questions: The Turing test, Searle's "Chinese Room" thought experiment, Ethical issues in AI *[Fundamental issues in intelligent systems]*
Problem spaces *[Search and constraint satisfaction]*
The role of heuristics *[Fundamental issues in intelligent systems]*
The nearest neighbor algorithm *[Machine learning and neural networks]*
Best-first search (generic best-first, Dijkstra's algorithm, A*, admissibility of A*) *[Search and constraint satisfaction]*
Two-player games (minimax search, alpha-beta pruning) *[Search and constraint satisfaction]*
Constraint satisfaction (backtracking and local search methods) *[Search and constraint satisfaction]*
Modeling the world *[Fundamental issues in intelligent systems]*
Review of propositional and predicate logic *[Knowledge representation and reasoning]*
Resolution and theorem proving *[Knowledge representation and reasoning]*
Structured representation: Frames and objects, Description logics, Inheritance systems *[Advanced knowledge representation and reasoning]*
Bayes theorem *[Knowledge representation and reasoning]*
Probabilistic reasoning *[Knowledge representation and reasoning]*

Uncertainty: Probabilistic reasoning, Bayesian nets, Fuzzy sets and possibility theory, Decision theory *[Advanced knowledge representation and reasoning]*
Deterministic and stochastic grammars *[Natural language processing]*
Parsing algorithms *[Natural language processing]*
Language translation *[Natural language processing]*
Definition and examples of machine learning *[Machine learning and neural networks]*
Supervised learning *[Machine learning and neural networks]*
Learning decision trees *[Machine learning and neural networks]*
Learning belief networks *[Machine learning and neural networks]*
Learning neural networks *[Machine learning and neural networks]*

AIAD Agentes e IA Distribuída (4/1)
Definition of agents *[Agents]*
Software agents, personal assistants, and information access: Collaborative agents, Information-gathering agents Agent theory: Commitments, Intentions, Decision-theoretic agents, Markov decision processes (MDP) *[Agents]*
Believable agents (synthetic characters, modeling emotions in agents) *[Agents]*
Agent architectures: Simple reactive agents, Reactive planners, Layered architectures, Example architectures and applications *[Agents]*
Successful applications and state-of-the-art agent-based systems *[Agents]*
Introduction to robotic agents *[Agents]*
Reasoning on action and change: Situation calculus, Event calculus, Ramification problems *[Advanced knowledge representation and reasoning]*
Multi-agent systems: Economically inspired multi-agent systems, Collaborating agents, Agent teams, Agent modeling, Multi-agent learning *[Agents]*
The role of middleware and support tools *[Wireless and mobile computing]*
Agent communication platforms and languages *[Multi-agent systems, cooperation and negotiation]*
Agent coordination *[Multi-agent systems, cooperation and negotiation]*
Agent cooperation and competition *[Multi-agent systems, cooperation and negotiation]*
Negotiation protocols *[Multi-agent systems, cooperation and negotiation]*
Ontologies *[Multi-agent systems, cooperation and negotiation]*
E-Business in Multi-Agent Systems applications *[Multi-agent systems, cooperation and negotiation]*
Mobile Internet protocol *[Wireless and mobile computing]*
Software package support for mobile and wireless computing *[Wireless and mobile computing]*
Explanation based learning *[Machine learning and neural networks]*
Mobile agents *[Agents]*
Learning theory *[Machine learning and neural networks]*
Learning agents *[Agents]*
Unsupervised learning *[Machine learning and neural networks]*
Reinforcement learning *[Machine learning and neural networks]*
Software characters and intelligent agents *[HCI aspects of collaboration and communication]*

Rob Robótica (4/2)
Overview: State-of-the-art robot systems, Planning vs. reactive control, Uncertainty in control, Sensing, World models *[Robotics]*
Agent based control architectures *[Robotics]*
Hybrid architectures *[Robotics]*
Navigation and control *[Robotics]*
Robot programming *[Robotics]*
Configuration space *[Robotics]*
Planning *[Robotics]*
Planning and robotics *[AI planning systems]*
Planning and execution *[AI planning systems]*
Static world planning systems *[AI planning systems]*

Planning as search *[AI planning systems]*
Propositional planning *[AI planning systems]*
Operator-based planning *[AI planning systems]*
Extending planning systems (case-based, learning, and probabilistic systems) *[AI planning systems]*
Definition and examples of planning systems *[AI planning systems]*
Learning plans *[AI planning systems]*

CIA Complementos de Inteligência Artificial (5/1)
Constraint satisfaction (backtracking and local search methods) *[Search and constraint satisfaction]*
Constraint Logic Programming *[Search and constraint satisfaction]*
Tools for Constraint Logic Programming *[Search and constraint satisfaction]*
Genetic algorithms *[Advanced search]*
Nonmonotonic reasoning: Nonclassical logics, Default reasoning, Belief revision, Preference logics, Integration of knowledge sources, Aggregation of conflicting belief *[Advanced knowledge representation and reasoning]*
Knowledge representation for diagnosis, qualitative representation *[Advanced knowledge representation and reasoning]*
Knowledge based systems (architectures) *[Advanced knowledge representation and reasoning]*
Modal based vs. diagnostic based systems *[Advanced knowledge representation and reasoning]*
Shells and Expert Systems examples *[Advanced knowledge representation and reasoning]*
Data clustering algorithms *[Machine learning and neural networks]*
Inductive Logic Programming *[Machine learning and neural networks]*
Advanced algorithms for Neural Networks *[Machine learning and neural networks]*

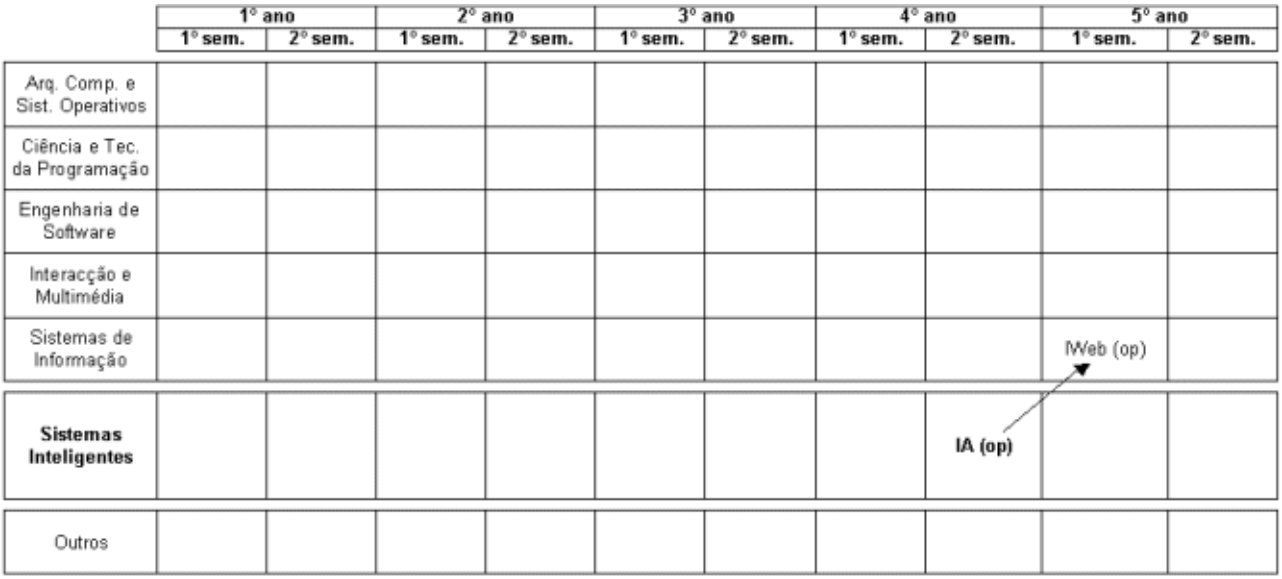
LNTA Linguagem Natural e Tradução Automática (5/1)
Language translation phases (lexical analysis, parsing, code generation, optimization) *[Introduction to language translation]*
Machine-dependent and machine-independent aspects of translation *[Introduction to language translation]*
Parsing algorithms *[Natural language processing]*
Deterministic and stochastic grammars *[Natural language processing]*
Language translation *[Natural language processing]*
Corpus-based methods *[Natural language processing]*
Information retrieval *[Natural language processing]*
Speech recognition *[Natural language processing]*

vii.3 LEEC

vii.3.1 Conteúdo

IA2 Inteligência Artificial (4/2)
Métodos de resolução de problemas. Representação de conhecimento. Métodos de raciocínio sobre conhecimento. Introdução ao processamento da linguagem natural. Aprendizagem simbólica e redes neuronais. Introdução à Programação Lógica

vii.3.2 Análise de Dependências



Análise das Dependências para Sistemas Inteligentes (LEEC)

vii.3.3 Comentários e Recomendações

A disciplina de IA inclui o uso da linguagem Prolog, que é ensinada aos alunos nesta mesma disciplina. Verifica-se que os alunos possuem reduzida prática de programação, e consequente falta de motivação neste domínio, facto que dificulta o ensino de uma nova linguagem.

vii.3.4 Disciplinas

IA2 Inteligência Artificial (4/2)
History of artificial intelligence *[Fundamental issues in intelligent systems]*
Philosophical questions *[Fundamental issues in intelligent systems]*
Fundamental definitions: Optimal vs. human-like reasoning, Optimal vs. human-like behavior *[Fundamental issues in intelligent systems]*
Philosophical questions: The Turing test, Searle's "Chinese Room" thought experiment, Ethical issues in AI *[Fundamental issues in intelligent systems]*
Problem spaces *[Search and constraint satisfaction]*
The role of heuristics *[Fundamental issues in intelligent systems]*
The nearest neighbor algorithm *[Machine learning and neural networks]*
Best-first search (generic best-first, Dijkstra's algorithm, A*, admissibility of A*) *[Search and constraint satisfaction]*
Two-player games (minimax search, alpha-beta pruning) *[Search and constraint satisfaction]*
Constraint satisfaction (backtracking and local search methods) *[Search and constraint satisfaction]*
Modeling the world *[Fundamental issues in intelligent systems]*
Review of propositional and predicate logic *[Knowledge representation and reasoning]*
Resolution and theorem proving *[Knowledge representation and reasoning]*
Structured representation: Frames and objects, Description logics, Inheritance systems *[Advanced knowledge representation and reasoning]*
Bayes theorem *[Knowledge representation and reasoning]*
Probabilistic reasoning *[Knowledge representation and reasoning]*
Uncertainty: Probabilistic reasoning, Bayesian nets, Fuzzy sets and possibility theory, Decision theory *[Advanced knowledge representation and reasoning]*
Deterministic and stochastic grammars *[Natural language processing]*
Parsing algorithms *[Natural language processing]*
Language translation *[Natural language processing]*
Definition and examples of machine learning *[Machine learning and neural networks]*

Supervised learning *[Machine learning and neural networks]*
Learning decision trees *[Machine learning and neural networks]*
Learning belief networks *[Machine learning and neural networks]*
Learning neural networks *[Machine learning and neural networks]*

viii. Conclusões

Fica para trabalho futuro tratar os pré-requisitos entre as Unidade de Conhecimento e verificar se os conteúdos das disciplinas os respeitam. Os objectivos nas fichas de disciplinas necessitam ainda de ser tratados devidamente uma vez que, nesta altura, são quase sempre os que decorrem das unidades CC2001 associadas às disciplinas e não são ainda apresentados nesta versão do relatório. Para os casos em que ainda não foi feito, podem ainda ser anotados os tópicos associados a uma dada disciplina com o nível de profundidade a que devem ser tratados.

Os relatórios apresentados de seguida pretendem ajudar na análise dos conteúdos das fichas de disciplina:

- Unidades de Conhecimento sem disciplina
- Unidades de Conhecimento em várias disciplinas agrupados por unidade
- Unidades de Conhecimento em várias disciplinas agrupados por área de secção
- Tópicos sem disciplina
- Tópicos em várias disciplinas
- Unidades de Conhecimento por disciplina na Ficha de Disciplina

Voltar ao Índice do Relatório

Documento versão 1.08, 18/07/2013, 11:14:46.

Trabalho organizado por: AAS, AMA, APM, APR, GTD, JCL, MCR.

João Correia Lopes (jlopes@fe.up.pt).
Page last modified on: Mon Jul 29 12:48:55 2002