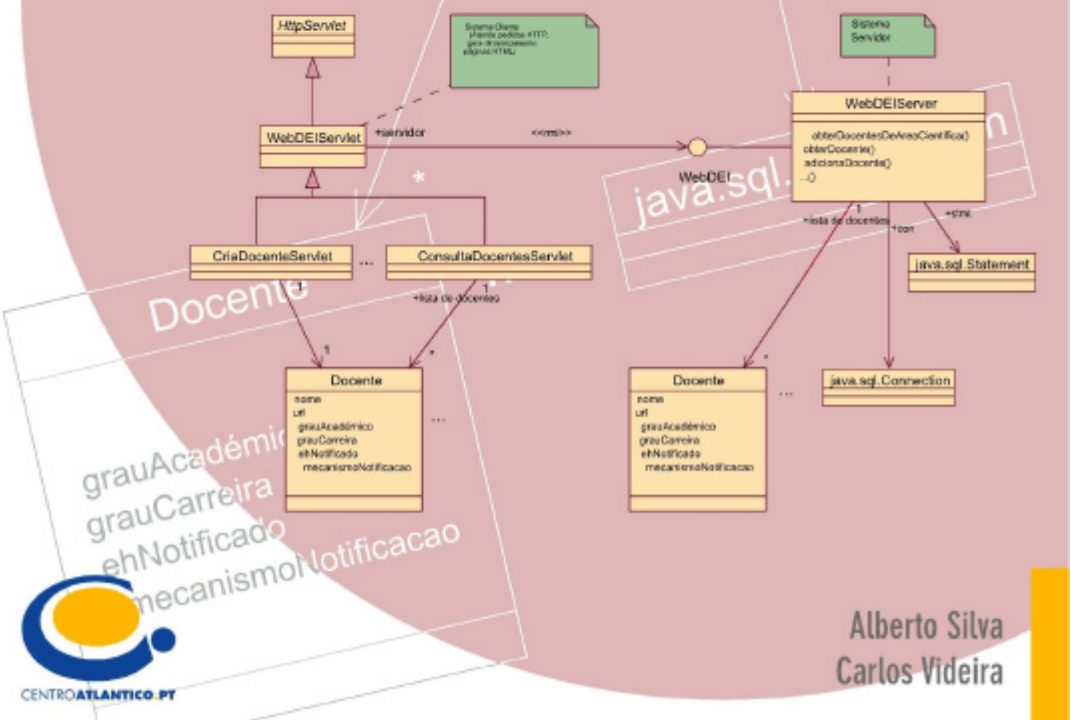
UML

METODOLOGIAS E FERRAMENTAS

CASE



Alberto Silva
Carlos Videira

CENTROATLANTICO.PT

Alberto Manuel Rodrigues da Silva
Carlos Alberto Escaleira Videira

UML, Metodologias e Ferramentas CASE

Linguagem de Modelação UML, Metodologias e Ferramentas CASE na Concepção e Desenvolvimento de Software

Edições Centro Atlântico
Portugal/2001

UML, Metodologias e Ferramentas CASE

Colecção: Tecnologias
Autores: Alberto Manuel Rodrigues da Silva
Carlos Alberto Escaleira Videira
Direcção gráfica: Centro Atlântico
Capa: Paulo Buchinho

Ap. 413 - 4760 V. N. Famalicão
Porto - Lisboa
Portugal
Tel. 808 20 22 21

geral@centroatlantico.pt
www.centroatlantico.pt

Fotolitos: Centro Atlântico
Impressão e acabamento: Inova
1ª edição: Abril de 2001

ISBN: 972-8426-36-4
Depósito legal: 164.544/01

À Graça, Joana e João Alberto

Alberto Silva

À Elsa, Sofia e Guilherme

Carlos Videira

Peça, gratuitamente, os ficheiros com as soluções dos exercícios ímpares deste livro

Receba gratuitamente, por e-mail, os ficheiros com as soluções dos exercícios ímpares deste livro, para poder comparar com as suas respostas. Para tal, envie a cópia da factura deste livro para o Centro Atlântico, para o e-mail,

geral@centroatlantico.pt

ou por correio para,

Centro Atlântico
Ap. 413
4760 V. N. Famalicão

Prefácio

Objectivos, Contexto e Motivação

O livro “UML, Metodologias e Ferramentas CASE” aborda tópicos importantes para a generalidade dos intervenientes nas actividades enquadradas na engenharia de software, designadamente as problemáticas (1) das linguagens de modelação de software, (2) do processo e das metodologias de desenvolvimento de software, e (3) das ferramentas CASE de suporte à modelação e ao próprio desenvolvimento. Pretende dar uma panorâmica abrangente sobre estes três aspectos de forma integrada e coerente. Embora o foco do livro seja nas fases de concepção de sistemas de software, discute o seu enquadramento de modo mais lato em áreas como o planeamento estratégico de sistemas de informação; as arquitecturas de sistemas de informação; ou mesmo a engenharia de software.

O livro explica a necessidade da modelação no desenvolvimento de software, o que é o UML (*Unified Modeling Language*), como aplicar o UML no contexto mais abrangente das metodologias e processos de desenvolvimento, e como usar ferramentas CASE de forma a maximizar e automatizar algumas das tarefas relacionadas com a modelação, por exemplo, produção e gestão de documentação, geração de código, geração de esquemas de dados, *reverse engineering*, *round-trip engineering*, mecanismos de extensão, etc.

A aprendizagem e adopção dos temas abordados neste livro constituem uma vantagem decisiva para os intervenientes que os adoptarem consistentemente. Entre outros, salientamos os seguintes benefícios: melhor documentação dos sistemas e dos respectivos artefactos; aplicação de técnicas de modelação orientadas por objectos, mais fáceis de entender; reutilização desde as fases preliminares da concepção até à implementação; rastreabilidade dos requisitos ao longo de todo o processo; facilidade de comunicação entre todos os intervenientes envolvidos

no processo; melhorias significativas em factores como sejam flexibilidade e produtividade; melhor gestão de requisitos; avaliação e manutenção de sistemas mais facilitadas. Estas características são naturalmente interdependentes entre si; por exemplo, uma maior qualidade da documentação produzida possibilita uma melhor comunicação entre os intervenientes de um projecto, ou uma melhor manutenção entre eles.

Todavia, os assuntos tratados neste livro são difíceis de adoptar nas organizações, por inúmeras razões. Antes de mais porque o ritmo de inovação tecnológica nesta área da engenharia tem-se processado a um ritmo particularmente intenso.

A segunda razão deve-se ao facto dos tópicos abordados neste livro exigirem uma formação significativa e principalmente uma adequada e correspondente actuação. Não basta dominar um conjunto alargado de conceitos e notações para especificar software, mas é fundamental aprender a aplicá-los de forma consistente, repetida e sistemática; adaptá-los às condicionantes e realidades de cada empresa, ou de cada projecto em particular; e ainda partilhar técnicas e métodos entre todos os indivíduos da empresa, ou de cada projecto, para que a comunicação entre todos os intervenientes seja maximizada e eficiente.

A terceira razão, consequência das razões anteriormente referidas, é o facto de ser oneroso a adopção efectiva e produtiva (dos tópicos abordados neste livro) no seio das empresas. Oneroso em termos do tempo inicial que é necessário despender em formação, em termos da "resistência à mudança", assim como o investimento necessário na selecção e aquisição de ferramentas CASE que potenciem significativamente as suas vantagens.

Este livro surge na sequência da experiência dos autores em actividades de investigação, mas principalmente em actividades de consultoria e de docência nas áreas de engenharia de software e de sistemas de informação.

Os temas abordados neste livro são na sua maioria influenciados pelo trabalho de unificação e de evangelização dos "três amigos": Grady Booch, Ivar Jacobson e James Rumbaugh. Todavia, é da nossa exclusiva responsabilidade o estilo do livro, assim como a sua estrutura, conteúdo, exemplos e exercícios propostos (tal como as correspondentes

gralhas e omissões decorrentes!). O livro condensa e integra informação dispersa por alguns livros da área, em particular os seguintes títulos: *OMG Unified Modeling Language Specification* [OMG99], *The Unified Modeling Language User Guide* [Booch99], *The Unified Software Development Process* [Jacobson99], *Visual Modeling with Rational Rose 2000 and UML* [Quatrani00] e *The Rational Unified Process* [Kruchten00]. No entanto, há inúmeros aspectos que o livro propõe e discute de forma única, dificilmente encontrados em qualquer dos livros referidos.

A nível internacional, existe um número relevante de títulos nesta área; contudo, há reconhecidamente na língua Portuguesa uma lacuna muito significativa. Paralelamente, e em consequência da nossa experiência e responsabilidade de docência, supervisão e coordenação de trabalhos finais de curso e de investigação identificámos a necessidade e oportunidade de produzirmos este livro com vista a apoiar a aprendizagem da engenharia de software nos tópicos referidos.

A temática tratada neste livro é abrangente e a sua profundidade é, propositadamente, de nível intermédio. Ínumeros assuntos poderão ser analisados e aprofundados complementarmente, entre os quais se destacam a título de exemplo os seguintes: arquitecturas de sistemas de software [Hofmeister99]; processos de negócio em contextos organizacionais [Penker00]; padrões de análise [Fowler96]; padrões de desenho em infra-estruturas de software (*frameworks*) [Souza99]; modelação de dados [Muller00]; modelação de aplicações segundo o paradigma dos agentes de software [Odell00], modelação de aplicações de tempo real [Selic94], ou modelação de aplicações interactivas [Nunes99]. Todos estes tópicos são importantes nos seus respectivos contextos de aplicação; muitos são alvo de intensa actividade de estudo e investigação. Todos eles apresentam, contudo, um denominador comum: baseiam-se no conhecimento introduzido, apresentado e discutido neste livro.

Audiência do Livro

O livro pretende servir como referência de suporte a um número restrito de disciplinas de nível de ensino superior na área de sistemas de informação. Consequentemente, o livro adopta um estilo tendencialmente pedagógico através da apresentação e discussão de exemplos, da narrativa de histórias e factos reais, ou pela proposta de exercícios académicos.

O primeiro perfil de leitores deste livro vai directamente para os alunos de licenciatura e de cursos de pós-graduação em engenharia informática ou em informática de gestão. Pressupõe-se que os leitores já "as-bem" implementar aplicações informáticas; e que neste livro procuram aprender a reflectir sobre o processo de desenvolvimento de software, e aprender técnicas e práticas consistentes e sistemáticas para o realizar.

Adicionalmente, este livro é relevante para um número mais alargado de leitores, em particular para investigadores, gestores informáticos, responsáveis pelo processo de desenvolvimento de software, analistas-programadores, e outros que necessitem de especificar de forma mais ou menos detalhada sistemas de software.

O livro pressupõe um conjunto de pré-requisitos que o leitor deverá possuir para o poder usufruir devidamente. É suposto o leitor possuir um conhecimento razoável sobre as bases da informática e dos sistemas de computadores, tais como noções essenciais de programação, de bases de dados e de sistemas operativos.

Organização do Livro

O livro encontra-se organizado em 4 partes, 14 capítulos e 2 apêndices conforme se resume de seguida.

A Parte 1 (INTRODUÇÃO E VISÃO GERAL) apresenta os conceitos gerais, visão histórica e enquadramento da realização deste livro. Inclui os capítulos 1, 2 e 3.

A Parte 2 (LINGUAGEM DE MODELAÇÃO UML) é constituída por 6 capítulos complementares, sendo que o Capítulo 4 dá a visão histórica e geral do UML e o Capítulo 9 descreve sucintamente alguns aspectos

considerados “avançados”, não essenciais para o leitor que apenas pretende usar e aplicar as características básicas do UML. Os restantes capítulos (Capítulos 5, 6, 7 e 8) constituem o centro desta parte do livro e deverão ser lidos de forma sequencial conforme proposto.

A Parte 3 (METODOLOGIAS DE DESENVOLVIMENTO DE SOFTWARE) apresenta a problemática geral das metodologias e processos de desenvolvimento de software, com exemplos concretos baseados em duas propostas reais de metodologias, o RUP e o ICONIX, descritos respectivamente nos Capítulos 10 a 11.

A Parte 4 (FERRAMENTAS CASE) apresenta a problemática das ferramentas CASE descrevendo o seu significado, evolução histórica e discutindo mecanismos de caracterização e avaliação (Capítulo 12). São apresentadas e analisadas duas ferramentas CASE, o Rose da Rational e o System Architect da Popkin, respectivamente nos Capítulos 13 e 14.

No Apêndice A (“Guia de Recursos Electrónicos”) apresenta-se de modo classificado um conjunto significativo de recursos electrónicos sobre os temas abordados neste livro.

No Apêndice B (“Glossário, Siglas e Abreviaturas”) apresentam-se três tabelas com informação relativa ao glossário, as siglas, e as abreviaturas adoptadas ao longo de todo o livro.

Em “Referências” listam-se, por ordem alfabética, todas as referências bibliográficas utilizadas ao longo do livro.

Por fim, inclui-se o “Índice Remissivo” que constitui um mecanismo típico de trabalho e de consulta neste género de literatura.

Notação Adoptada

Ao longo do livro são adoptadas genericamente as seguintes regras de notação textual:

- Nomes e expressões em inglês são escritas em itálico. As excepções são expressões vulgarmente adoptadas para o Português (e.g., software, bit), expressões intensamente usadas ao longo do texto (e.g., Internet, Web, applet, standard), ou nomes de empresas ou produtos de origem anglo-saxónica (e.g., MS-Word, Rational Rose).
- Frases e expressões que se pretendam destacar são escritas com ênfase (i.e., negrito).
- Exemplos de código, pseudo código, nomes de classes, ou endereços electrónicos são apresentados numa fonte de tamanho fixo (i.e., `Courier`).

Os exemplos apresentados neste livro aparecem enquadrados por uma moldura correspondente, conforme ilustrado neste mesmo parágrafo.

Há ao longo do livro um cuidado particular na devida introdução dos inúmeros conceitos que o mesmo analisa e discute. De forma a facilitar a identificação desses conceitos, colocamos na margem esquerda do respectivo texto a marca visual “Conceito” conforme apresentado neste parágrafo. Recomenda-se ao leitor a utilização do índice remissivo para consultar a definição de qualquer dos conceitos tratados neste livro.

Conceito

Por fim, relativamente à representação de diagramas será utilizada, sempre que for adequado, e por razões óbvias, a linguagem UML.

Agradecimentos

Um agradecimento muito especial à minha família por todo o amor e suporte que tive para poder realizar mais este trabalho, bem como pelas inúmeras horas roubadas ao seu convívio.

Um agradecimento também aos colegas do Judo Clube Portugal e outros amigos cujo convívio me proporcionou os momentos de relaxamento necessário para a produção deste livro.

Parte significativa da actividade que conduziu à realização deste livro foi desenvolvida no âmbito de duas instituições que procuram a excelência - o Departamento de Engenharia Informática do Instituto Superior Técnico e o Instituto de Engenharia de Sistemas e Computadores, às quais não posso deixar de endereçar o meu expresso agradecimento, bem como a todos os colegas e alunos com quem tive o privilégio de conviver, aprender e ensinar durante este período. Em particular, aos alunos da primeira e segunda edição da Pós-Graduação em Sistemas de Informação (POSI'1999 e POSI'2000) do Instituto Superior Técnico, com os quais ensaiei e testei uma parte preliminar deste livro; ao núcleo organizativo do POSI, nomeadamente aos Prof. José Tribolet e Prof. Paulo Guedes, pelo convite que me endereçaram; e ao meu monitor desses cursos, Eng. Miguel Goulão, com quem discuti alguns dos tópicos e exemplos apresentados.

Um agradecimento à editora Centro Atlântico, na pessoa do Dr. Libório Manuel Silva, pelo seu interesse imediato na publicação do livro e pela sua activa e persistente atitude de estar no nosso pequeno mercado nacional de literatura técnico-científica.

Por fim, um agradecimento a todos os colegas que de uma forma ou outra sugeriram, comentaram ou apenas criticaram partes preliminares deste trabalho, ou com quem simplesmente fui partilhando a "ideia" do livro.

Alberto Silva

Quero em primeiro lugar agradecer à minha família, pela sua dedicação, carinho e apoio incondicional, sem a colaboração da qual dificilmente teria participado neste projecto. Quero também agradecer aos meus amigos, de cujo convívio tive que prescindir para poder completar este livro.

Para a realização bem sucedida deste meu projecto foi também decisiva a contribuição de todos os meus colegas da Mentor IT, com os quais tenho abordado alguns temas que são explorados neste livro. A experiência adquirida nos vários projectos em que participei permitiram-me solidificar conhecimentos e sustentar algumas opiniões emitidas neste livro.

Um factor decisivo para a minha participação neste livro foi a experiência como docente, especialmente na Universidade Autónoma de Lisboa, onde tenho estado ligado a disciplinas relacionadas com os temas abordados neste livro. Nesse sentido, gostaria de agradecer ao Prof. José Luís Ferreira e ao Eng. Miguel Gonçalves toda a colaboração e incentivo que me têm dado, bem como o seu contributo em termos de algumas opiniões. Um agradecimento particular a todos os alunos das várias disciplinas que leccionei, pois o esforço de preparação das mêsmas contribuiu para a evolução do conteúdo de uma parte significativa deste livro.

Um agradecimento também para outros colegas com quem mantive, ao longo destes meses de trabalho, uma permuta de opiniões e críticas que me ajudaram a melhorar a qualidade da presente obra.

Finalmente, à Editora Centro Atlântico e ao Dr. Libório Manuel Silva deixo um agradecimento pelo seu interesse na publicação desta obra técnico-científica, valorizando a missão de educar para o futuro.

Carlos Videira

Contactos

Comentários técnicos, sugestões, pedidos de livros ou pedidos de esclarecimentos podem ser dirigidos ao Centro Atlântico (via *www.centroatlantico.pt* ou *geral@centroatl.pt*) que os encaminhará aos autores via correio electrónico se a sua colaboração for necessária.

Autores

Alberto Manuel Rodrigues da Silva é professor auxiliar no Departamento de Engenharia Informática do IST/UTL, investigador sénior no INESC e consultor informático em diferentes empresas e instituições. Tem um doutoramento em Engenharia Informática e Computadores pelo IST/UTL, um mestrado em Engenharia Electrotécnica e Computadores pelo IST/UTL e uma licenciatura em Engenharia Informática pela FCT/UNL. Lecciona actualmente cadeiras da área de Sistemas de Informação e de Engenharia de Software de nível licenciatura, pós-graduação e mestrado. Supervisiona a realização de vários trabalhos finais de curso e de teses de mestrado. Tem interesses profissionais e científicos em sistemas de informação distribuídos em larga escala e em aplicações Web; modelização de software, processos de desenvolvimento de software; e negócios suportados electronicamente. É autor de 2 livros técnicos e cerca de 30 artigos científicos em revistas, conferências e workshops nacionais e internacionais.

Carlos Alberto Escaleira Videira é actualmente *Consulting Manager* na *Mentor*IT, empresa de consultoria estratégica na área dos sistemas de informação, e assistente no Departamento de Ciências e Tecnologias da UAL. Desempenhou funções de coordenação na área de Infor-

mática em diferentes empresas e participou em diversos projectos como consultor. Tem um mestrado em Engenharia Electrotécnica e Computadores pelo IST/UTL e uma licenciatura em Engenharia Informática pela FCT/UNL. Lecciona actualmente cadeiras da área de Planeamento de Sistemas de Informação, Engenharia de Software e Negócios Electrónicos de nível de licenciatura e pós-graduação. Tem interesses profissionais e científicos em temas relacionados com Planeamento Estratégico de Sistemas de Informação, Engenharia de Software, Sistemas de Informação, Gestão de Projectos e Negócios Electrónicos.

Lisboa, Março de 2001

Alberto Manuel Rodrigues da Silva

Carlos Alberto Escaleira Videira

Índice

Parte 1 – Introdução e Visão Geral

Uma empresa de software de sucesso é aquela que consistentemente produz software de qualidade que vai ao encontro das necessidades dos seus utilizadores. Uma empresa que consegue desenvolver tal software, de forma previsível, cumprindo os prazos, com uma gestão de recursos, quer humanos quer materiais, eficiente e eficaz, é uma empresa que tem um negócio sustentado.

Grady Booch, James Rumbaugh, Ivar Jacobson. *The Unified Modeling Language User Guide*.

Fazer software não é uma tarefa fácil. Fazer software de qualidade é ainda mais difícil. A generalidade dos resultados obtidos ao longo do tempo têm sistematicamente apresentado padrões de baixa qualidade, de custos e prazos completamente ultrapassados. Neste aspecto, a indústria de software deve ser caso único na sociedade actual, pois

apesar da taxa de sucesso dos projectos ser relativamente baixa, o interesse das organizações pelo desenvolvimento de sistemas informáticos tem aumentado constantemente, não se vislumbrando qualquer alternativa. Tudo isto porque as organizações reconhecem que o recurso informação é estratégico e fonte de vantagens competitivas importantes.

O facto dos resultados dos projectos informáticos estarem normalmente abaixo das expectativas e dos diversos problemas que de forma consistente vêm ocorrendo desde o início da utilização das tecnologias de informação, torna extremamente relevantes as várias iniciativas que possam ser desenvolvidas com o objectivo de ultrapassar estes problemas. Sobretudo, vale a pena analisar os diversos esforços que foram efectuados ao longo do tempo, e perceber por que alguns não foram totalmente efectivos na resolução dos problemas, enquanto outros, bem sucedidos, são apontados como melhores práticas a aplicar sistematicamente.

Esta primeira parte do livro pretende dar um enquadramento das questões relacionadas com o desenvolvimento de software, de forma a "aguçar o apetite" dos leitores para os capítulos subsequentes do livro, onde são apresentadas várias ideias, técnicas, métodos e ferramentas que os autores deste livro acreditam que poderão desempenhar um papel decisivo na melhoria dos diversos problemas referidos na primeira parte.

Organização da Parte 1

O Capítulo 1, "Enquadramento e Conceitos Gerais", faz o enquadramento e define o âmbito do livro em questões mais vastas relacionadas com as tecnologias de informação, de forma a transmitir a mensagem ao utilizador que há questões importantes relacionadas com o desenvolvimento de software cuja resolução passa pela realização de actividades e aplicação de técnicas que saem fora do âmbito deste livro. Apresenta ainda os problemas que os sistemas de informação enfrentam actualmente e algumas definições que são relevantes para a compreensão do livro.

O Capítulo 2, “O Processo de Desenvolvimento de Software”, pretende fornecer ao leitor uma visão geral sobre as actividades relacionadas com o desenvolvimento de software, nomeadamente sobre a sua organização, sequência e objectivos a atingir. São ainda clarificados alguns conceitos relacionados com as etapas do desenvolvimento de software.

O Capítulo 3, “Evolução das Metodologias de Desenvolvimento de Software”, procura dar uma visão histórica de como o desenvolvimento de software foi encarado ao longo do tempo, na perspectiva da aplicação de metodologias e respectivas técnicas, e quais as principais motivações para os diversos saltos qualitativos que ocorreram.

Capítulo 1 - ENQUADRAMENTO E CONCEITOS GERAIS

Tópicos

- Introdução
- O Impacto das Tecnologias de Informação
- Produto e Processo
- Sistemas de Informação
- Arquitectura de Sistemas de Informação
- Objectivos do Desenvolvimento de Sistemas de Informação
- Problemas no Desenvolvimento de Sistemas de Informação
- Planeamento Estratégico de Sistemas de Informação
- Engenharia de Software
- Conclusão
- Exercícios

1.1 Introdução

O objectivo deste livro é apresentar a linguagem de modelação UML (Parte 2) e demonstrar a sua aplicação de forma a facilitar todo o desenvolvimento de software, quer seja directamente como técnica de modelação de software, quer seja na sua utilização em metodologias de desenvolvimento (Parte 3) ou em ferramentas de apoio (Parte 4).

De forma a compreender as principais razões por que muitos de nós, ligados à área académica e profissional das tecnologias de informação, acreditamos que o UML representa já actualmente um papel relevante no desenvolvimento de software, é importante enquadrar o leitor deste livro no que consideramos os principais problemas, objectivos e conceitos relacionados com os sistemas de informação e com o seu desenvolvimento. Neste primeiro capítulo, esta abordagem será efectuada de forma ainda muito genérica, e será concretizada nos dois capítulos seguintes.

É ainda importante que o leitor compreenda a relevância de outros conceitos e actividades, que devem ser aplicados no âmbito dos sistemas de informação, mas que não se encontram no âmbito deste livro; estamos a falar, por exemplo, das noções de arquitectura de sistemas de informação e do planeamento estratégico de sistemas de informação. São áreas que estão ao nível da concepção de sistemas de informação, com preocupações de natureza estratégica e que apenas serão brevemente equacionadas neste livro.

1.2 O Impacto das Tecnologias de Informação

É hoje em dia lugar comum ouvir-se falar da importância que a informática ocupa na nossa vida. O impacto e a rápida evolução ao longo dos últimos 40 anos das tecnologias relacionadas com os sistemas de informação tem colocado sucessivos desafios às empresas. De forma a tirar partido das potencialidades destas tecnologias, é necessário um grande investimento em software e hardware. Este impacto é visível não só nas grandes organizações de âmbito internacional, mas atinge também as pequenas e médias empresas.

Desde que surgiram, as tecnologias de informação potenciaram o aparecimento de novas indústrias, como sejam as consultoras de sistemas de informação ou as relacionadas com negócios na Internet, ou reforçaram a importância de outras, nomeadamente as ligadas à indústria de telecomunicações. Têm também provocado uma redefinição das responsabilidades e das interacções entre os parceiros da cadeia de valor de várias indústrias. Nos anos mais recentes, as tecnolo-

gias de informação têm mesmo posto em causa modelos tradicionais de fazer negócio.

Ao longo do tempo, o papel das tecnologias de informação nas organizações sofreu diversas alterações. Actualmente, as tecnologias de informação encontram-se na origem de mudanças significativas ao nível dos modelos de negócio das empresas, e constituem um elemento fundamental para a obtenção de vantagens estratégicas e competitivas. Por isso, a respectiva implementação nas organizações deve ser cuidadosamente planificada e estruturada, de modo a garantir o alinhamento com os objectivos estratégicos do negócio.

A implementação de sistemas de informação requer um investimento significativo (financeiro, tecnológico e de recursos humanos), pelo que estas intervenções deverão merecer o apoio e o comprometimento das administrações. A justificação destes volumes de investimento deve ser efectuada demonstrando qualitativamente e quantitativamente o seu valor estratégico e o impacto positivo nas organizações.

No entanto, muitos gestores não conseguem perceber o verdadeiro alcance de todas estas tecnologias, quer por questões de formação, quer pela sua anterior experiência com sistemas antiquados e obsoletos, que constituíam verdadeiros entraves à satisfação dos requisitos do negócio, e não funcionavam como potenciadores do seu crescimento. Por outro lado, os intervenientes da área de informática criaram no passado uma imagem muito técnica, pouco alinhada com as reais necessidades do negócio, o que contribuiu decisivamente para a não caracterização da informática como uma área estratégica dentro das empresas.

A progressiva importância que os sistemas de informação têm nas organizações pode ser constatada através de diversas situações:

- No passado era comum o responsável da informática depender hierarquicamente do director financeiro, enquanto este reportava directamente à administração. Pelo contrário, actualmente são cada vez menos as organizações em que esta situação se mantém, ficando a área de informática ao mesmo nível que os restantes departamentos e reportando directamente ao órgão que define as respectivas estratégias, a administração; a informática passa assim a ser considerada como uma área estratégica.
- A indústria de software, ou de forma mais geral todas as relacionadas com as tecnologias de informação, é actualmente uma das mais importantes em todo o planeta e uma das principais responsáveis pelo crescimento contínuo da economia mundial durante a última década. Este fenómeno é também visível ao nível das individualidades, já que o homem mais rico do mundo é actualmente um dos principais responsáveis pela maior empresa de software (estamos obviamente a falar de Bill Gates e da Microsoft).
- A crescente importância das empresas relacionadas com a Nova Economia (que de forma simplificada poderemos associar ao fenómeno Internet), cujas acções são transaccionadas nos Estados Unidos num bolsa de valores específica (Nasdaq).
- A importância destas empresas tem motivado a crescente preocupação dos governos em garantir o acesso livre ao mercado e a tentar evitar posições monopolistas. É o caso do presente litígio entre o governo americano e a Microsoft, onde assistimos à disputa em torno de questões por vezes pouco racionais; no entanto, e independentemente da nossa posição pessoal, o governo americano actua de forma semelhante à dos seus antecessores há algumas décadas atrás, em relação a empresas de outras indústrias chave, como eram na altura a do petróleo e do aço.

Muitos outros exemplos poderiam ser dados, mas a conclusão óbvia é que nos tornámos dependentes das tecnologias de informação, quer do ponto de vista pessoal quer profissional.

1.3 Produto e Processo

A importância das tecnologias de informação na nossa vida é sobretudo concretizada pelas funcionalidades que são implementadas ao nível do software, e que são disponibilizadas com o suporte de um conjunto de dispositivos diversos (hardware). O primeiro pode ser considerado o componente lógico dos sistemas de informação, o segundo o componente físico.

Não existe uma definição rigorosa e inequívoca de software. Diversos autores [Pressman2000, Schach1999] encaram o software como o resultado final de um processo, ao qual designam por “Engenharia de Software”. O que é um facto é que o software não é dádiva da natureza, nem é objecto de uma produção numa linha de montagem, realizada de forma perfeitamente automática, sem qualquer intervenção humana, criativa e subjectiva.

Quando falamos em "processo" esta palavra implica desde logo a definição de um conjunto de actividades uniformizadas, a aplicar sistematicamente, que se encontram agrupadas em fases. Cada uma destas fases tem os seus intervenientes, aos quais são atribuídas responsabilidades, que possui diversos *inputs* e que produz *outputs*. Do ponto de vista da garantia da qualidade do produto final (o software), é fundamental que o processo seja realizado segundo parâmetros que permitam também aferir a respectiva qualidade, isto é, não conseguiremos optimizar o resultado final sem uma preocupação no processo que o produz.

Se pensarmos que o desenvolvimento do software é um processo que deve ser baseado na aplicação de técnicas e práticas rigorosas, sistemáticas, eficientes e controláveis, podemos concluir que este se aproxima bastante de outras realizações humanas, como a construção de qualquer obra de engenharia civil (por exemplo, a construção da ponte Vasco da Gama em Lisboa). Daí o nome de "Engenharia de Software" precisamente como tentativa de trazer para esta actividade a preocupação da aplicação de técnicas de engenharia ao desenvolvimento de software, por exemplo, modelar antes de realizar; estimar

diversos factores antes de avançar; medir antes, durante e depois do produto realizado; analisar factores de risco.

Para além dos elementos já descritos, tal como nas outras engenharias, também a realização efectiva das funções de desenvolvimento de software pressupõe a utilização de ferramentas de apoio a todo o processo. O tempo em que o desenvolvimento era efectuado de forma completamente manual já não é razoável actualmente (tal como ninguém constrói uma casa, e muito menos uma ponte, unicamente à custa do seu esforço físico). As características destas ferramentas podem ter um impacto apreciável no produto final (bem como no processo), e a demonstração desse facto é um dos objectivos deste livro.

No entanto, é também importante esclarecer desde já que a produção de software encerra em si mesma alguma subjectividade, devido ao facto de ser realizada por seres humanos, que em diversos pontos podem introduzir factores resultantes da opinião pessoal (e que até certo ponto podem ser benéficos, pois a criatividade pode levar à produção de software com melhor aceitação e desempenho). Neste aspecto, o processo aproxima-se mais de uma actividade artística do que propriamente uma actividade de engenharia. É por isso que nós consideramos, tal como outros autores, que o ponto de equilíbrio correcto depende de cada caso, mas deve-se encontrar a meio caminho entre a aplicação de técnicas estruturadas (Engenharia) e introdução de factores de criatividade (Arte).

Actualmente, e num contexto social e económico em constante mudança, espera-se que o software seja capaz de evoluir a um ritmo que não ponha em causa o crescimento das organizações. São por isso fundamentais as seguintes características:

- Flexibilidade, enquanto capacidade de evolução face aos requisitos de negócio.
- Fiabilidade, o que implica que o número de problemas ocorrido seja reduzido e não ponha em causa o funcionamento das organizações.
- Implementação das necessidades das organizações
- Nível de desempenho adequado

- Facilidade de utilização, com uma interface amigável e intuitiva para o utilizador.

1.4 Sistemas de Informação

A visão mais tradicional sobre o conceito de **software** limita-se a considerá-lo como um conjunto de programas, constituído por blocos de código. Outros autores englobam ainda neste conceito a documentação de apoio que é produzida. No entanto, quando falamos actualmente do componente lógico que serve de suporte às necessidades das organizações, o conceito mais abrangente normalmente utilizado é o de sistemas de informação.

Tal como em muitas outras situações no domínio da informática, não existe uma definição formal e consensual deste conceito. Neste livro adoptaremos a seguinte definição: um **sistema de informação** é um conjunto integrado de recursos (humanos e tecnológicos) cujo objectivo é satisfazer adequadamente a totalidade das necessidades de informação de uma organização e os respectivos processos de negócio. Nesta definição o conceito **processo de negócio** pretende representar uma sequência de actividades, que processam vários *inputs* e produzem vários *outputs* e que possuem objectivos. Pode ser realizado por pessoas e/ou de forma automática. Exemplos de processos de negócio incluem as compras de matérias-primas, a contratação de um empregado ou a distribuição de produtos acabados.

Existem outras definições para o conceito de sistema de informação que enumeram os respectivos componentes, nomeadamente pessoas, hardware, software, redes e dados, sempre numa perspectiva integrada, e de modo a suportar e melhorar as operações diárias do negócio, bem como a satisfazer as necessidades de informação dos gestores [O'Brien00]. Finalmente, de referir que alguns autores não consideram a parte de processos manuais como fazendo parte do sistema de informação.

Os sistemas de informação são actualmente considerados essenciais para suportar adequadamente estratégias de globalização e de reengenharia de processos de negócio e para a obtenção de vantagens

competitivas, com impacto ao nível da redução de custos, estratégias de diferenciação e/ou de inovação, promovendo e facilitando as relações e negócio com parceiros e clientes. É objectivo fundamental dos sistemas de informação garantir o alinhamento das tecnologias da informação com os objectivos estratégicos do negócio.

O impacto dos sistemas de informação nas organizações é inegável e inevitável. Uma das mais antigas classificações de sistemas de informação foi proposta por Anthony em 1965 [Anthony65]. Esta classificação agrupava os sistemas de informação em função do nível das actividades de gestão dentro da organização no qual o software tem impacto:

- Operacional, onde se incluíam todos os sistemas de informação que suportavam directamente as operações do dia-a-dia. Estamos a falar sobretudo de operações que implicam alterações na informação.
- Táctico, que inclui as funcionalidades de análise de informação, sobretudo orientadas para suportar o processo de tomada de decisões com impacto na gestão de curto prazo.
- Estratégico, essencialmente preocupado com questões de planeamento, em que o impacto se situa temporalmente no médio e longo prazo.

Tipo de Sistemas	**Exemplos**
Operacionais	Facturação, Controlo de encomendas, Contabilidade geral, Controle de Stocks, Salários
Tácticos	Análise de vendas, Controle orçamental, Contabilidade analítica, Gestão do inventário, Análise da qualidade
Estratégicos	Previsão de vendas, Planeamento da alocação da produção, Planeamento recursos humanos, Previsão de receitas e custos, Modelação financeira

Tabela 1.1: Exemplos de sistemas de informação segundo a classificação de Anthony.

Muitas outras classificações existem, segundo parâmetros variados, mas a sua apresentação sai fora do âmbito deste livro.

1.5 Arquitectura de Sistemas de Informação

A crescente complexidade dos sistemas de informação e a dificuldade de apresentação da sua estrutura aos diversos interessados, incluindo utilizadores e informáticos, motivou durante a década de 80 e inícios da década de 90 um conjunto de esforços no sentido de formalizar e uniformizar a respectiva apresentação, de modo a garantir, adicionalmente, a integração dos diversos componentes de informação da organização.

Em 1987, John Zachman publicou o artigo "*A Framework for Information Systems Architecture*" [Zachman87], em que introduzia o conceito de arquitectura de sistemas de informação. As ideias propostas resultaram de conhecimentos e experiências de outras disciplinas mais antigas (arquitectura, engenharia da produção) e rapidamente se tornaram numa referência para todos aqueles que têm algum interesse no tema da arquitectura de sistemas de informação. Infelizmente, e apesar da relevância do tema, muitos destes conceitos continuam desconhecidos da maioria do público informático.

De acordo com este autor, a **arquitectura** é o "conjunto de representações descritivas (modelos) relevantes para a descrição de um objecto, de forma a que este possa ser construído de acordo com os requisitos (de qualidade) e mantido ao longo da sua vida útil".

Esta definição é consideravelmente genérica e informal e não indica o âmbito do termo arquitectura; de facto, no caso da abordagem proposta por Zachman, ela refere-se quer aos sistemas de informação quer à empresa, uma vez que o mesmo modelo apresenta relativamente a cada conceito a perspectiva do negócio e dos sistemas de informação.

O *Framework de Zachman* é uma estrutura lógica de classificação e apresentação dos modelos de uma organização relevantes para a respectiva gestão, bem como para o desenvolvimento dos seus sistemas, e pode ser observado na Figura 1.1. Nesta perspectiva, modelar um sistema significa determinar e representar um conjunto de informação, sobre

vários tópicos (colunas da matriz), relevante para vários intervenientes (linhas da matriz).

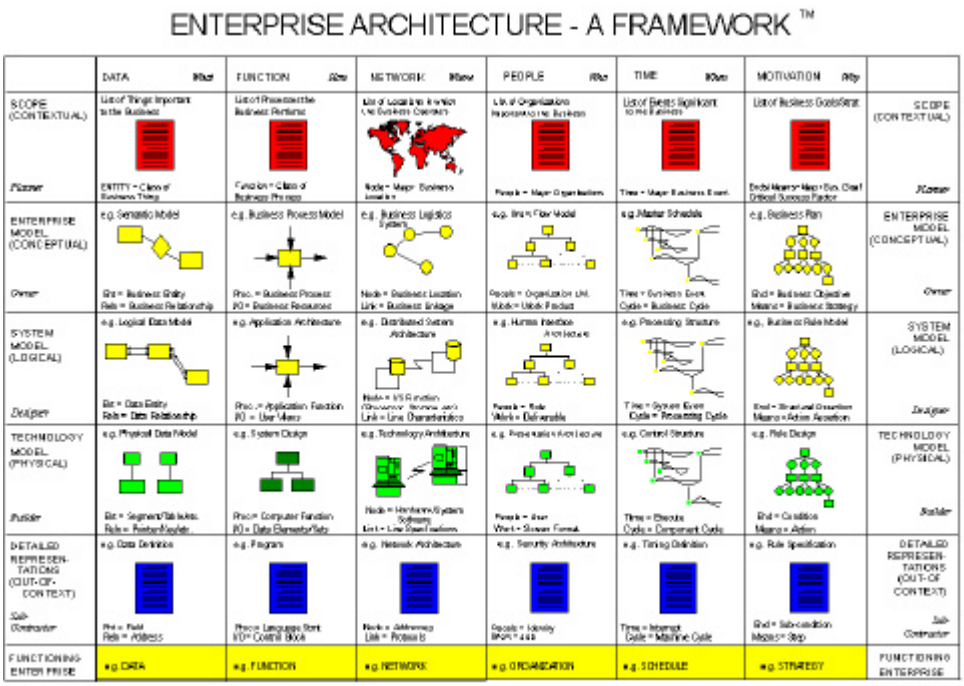


Figura 1.1: Framework de Zachman.

Este diagrama apresenta a relação entre as diferentes funções que podem ser identificadas na organização, e a visão e detalhe que têm (e precisam de ter) sobre os diversos objectos e conceitos da organização. Assim, são considerados cinco perfis de intervenientes que se relacionam com o sistema:

- *Planner*, responsável pelo planeamento estratégico da organização.
- *Owner*, responsável pela operação do negócio.
- *Designer*, responsável pela elaboração da especificação funcional do sistema.
- *Builder*, responsável pela elaboração da especificação técnica do sistema.
- *Sub-contractor*, responsável pela especificação detalhada e construção do sistema.

Os dois primeiros níveis são tipicamente utilizadores do sistema e relacionados com as áreas do negócio, enquanto os três últimos são intervenientes com perfil informático. À medida que se desce na hierarquia, aumenta o nível de detalhe a que a análise e a modelação têm que ser efectuadas. Cada um destes perfis tem uma visão diferente sobre um conjunto de factores analisados pelo *framework*, designadamente:

- Qual a constituição do sistema (*What*) - os dados?
- Como é que o sistema funciona (*How*) – as funções?
- Onde está localizado o sistema (*Where*) – as relações e as redes?
- Quem são os interessados no sistema (*Who*) – as pessoas?
- Quando ocorrem factos relevantes no sistema (*When*) – o tempo?
- Porque é que o sistema funciona assim (*Why*) – as motivações?

Este tipo de abordagem muito estruturada permite utilizar um único modelo para simplificar a compreensão e comunicação sobre a visão da organização; dar ênfase à análise de variáveis independentes; e manter uma perspectiva disciplinada sobre relações necessárias para preservar a integridade dos conceitos da organização. Pode ser utilizada nas diferentes fases do ciclo de desenvolvimento de sistemas de informação, desde o planeamento estratégico até ao desenho técnico detalhado.

Uma outra abordagem alternativa baseia-se no *Framework de Index* [Wurman97], e considera que a arquitectura de sistemas de informação é um conjunto integrado e consistente de componentes, que são definidos de forma a garantir o respectivo alinhamento com os objectivos de negócio, e por isso são suportados por todos os elementos da organização. Estes componentes encontram-se normalmente organizados em quatro grandes blocos:

- Arquitectura aplicacional: conjunto de sistemas e aplicações necessários para suportar os objectivos de negócio da organização.
- Arquitectura tecnológica: componentes de infra-estrutura e máquinas necessários para suportar as funcionalidades e requisitos das aplicações identificadas.
- Arquitectura de dados: conceitos e entidades necessárias à execução dos processos de negócio da organização.

- Arquitectura organizacional: estrutura de recursos humanos necessária para suportar adequadamente os restantes componentes dos sistemas de informação.

A definição destes componentes deve obedecer a uma sequência lógica, que tem a ver com as precedências e as interligações que existem entre eles. Como o componente que suporta os objectivos de negócio são as aplicações, estas devem ser identificadas em primeiro lugar, em paralelo com os conceitos (dados) que gerem. As componentes tecnológica e organizacional serão as últimas a ser definidas, de forma a suportarem adequadamente as restantes.

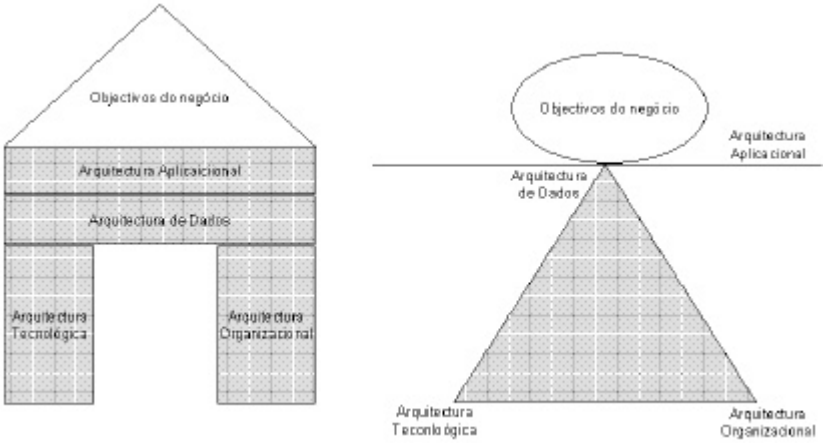


Figura 1.2: Representações da Arquitectura de Sistemas de Informação.

A Figura 1.2 ilustra de uma forma esquemática e simbólica a importância da definição de uma arquitectura estável em que os diversos componentes estão relacionados entre si de forma equilibrada. A parte esquerda da Figura 1.2 pretende representar uma arquitectura estável, em que os componentes estão solidamente integrados, ao contrário do que acontece na parte direita da mesma figura, em que a arquitectura é claramente instável e o seu equilíbrio deficiente.

1.6 Objectivos do Desenvolvimento de Sistemas de Informação

Em 1983, Robert Block definiu um sistema de informação bem sucedido como sendo aquele que é produzido dentro do prazo e nos custos estimados; é fiável (sem erros e disponível quando necessário) e pode ser mantido facilmente e a baixo custo; responde adequadamente aos requisitos definidos; e satisfaz os utilizadores. Esta definição, demasiado restrita, leva à conclusão natural de que poucos serão os sistemas que respeitam estes requisitos [Block83].

Ao longo do tempo, o papel do software e dos sistemas de informação nas organizações tem evoluído de forma a posicionar-se cada vez mais como factor estratégico e competitivo. Nos primórdios da computação (há apenas 50 anos atrás), o software era utilizado sobretudo para a resolução de problemas de cálculo relacionados com questões militares (por exemplo, cálculo das trajectórias de projecteis). Os primeiros computadores com aplicações de natureza comercial eram utilizados pelas grandes organizações com o objectivo de automatizar algumas das etapas dos processos de negócio e desta forma reduzir custos. A partir deste momento a importância e impacto dos sistemas de infor-mação nas organizações não tem parado de crescer, e podemos carac-terizá-la resumidamente de acordo com o apresentado na Figura 1.3.

Figura 1.3: Factos relevantes na evolução dos sistemas de informação.

Outras classificações foram elaboradas, nomeadamente a de Primozic [Primozic90], que identifica cinco grandes ondas de inovação, de acordo com a evolução das tecnologias de informação e os benefícios crescentes que oferecem às organizações.

Onda de Inovação	**Utilização Funcional**	**Impacto na Organização**
Reduzir Custos	Administrativas	Gestão de processos
Potenciar Investimentos	Financeiras, Produção	Gestão de recursos
Melhorar e aumentar produtos e serviços	Marketing, Distribuição, Apoio ao Cliente	Crescimento e aumento da quota de mercado
Melhorar a eficácia das decisões	Decisões Estratégicas	Reengenharia da organização
Atingir o consumidor	Funcionalidades nos computadores dos clientes	Reestruturação da indústria

Tabela 1.2: Ondas de Inovação de Primozic.

Independentemente destas classificações, existe um conjunto de razões que levam as organizações a investir em sistemas de informação e que podemos indicar de seguida, de forma resumida:

- Reduzir custos operacionais, através da automatização e reformulação dos processos de negócio.
- Satisfazer requisitos de informação dos utilizadores.
- Contribuir para a criação de novos produtos e serviços.
- Melhorar o nível de serviço prestado aos clientes actuais e facilitar a aquisição de novos clientes.
- Melhorar e automatizar a relação com os parceiros de negócio.
- Melhorar o desempenho de pessoas e máquinas.

1.7 Problemas no Desenvolvimento de Sistemas de Informação

Historicamente, o software tem apresentado de forma sistemática e contínua os mesmos problemas. As razões que no passado justificaram a adopção de métodos de trabalho mais estruturados continuam a verificar-se e por isso somos levados a concluir que estas iniciativas não vieram, afinal, resolver de todo os problemas. Se pensarmos no impacto na organização, estes podem ser essencialmente agrupados em três níveis:

- Falta de qualidade, traduzida na satisfação incompleta dos requisitos e nos problemas que se verificam após a instalação do produto.
- Desvios dos prazos previamente estabelecidos para o desenvolvimento de software.
- Custos previamente definidos para o desenvolvimento de software largamente ultrapassados.

A Tabela 1.3 ilustra como ao longo do tempo os diversos problemas têm existido de forma contínua, e independentemente das iniciativas que têm surgido, estas não eliminaram de forma alguma o problema.

1979	Em 57 projectos, 46% estavam atrasados (média de 7 meses) e 59% encontravam-se acima do orçamento [Lehman79].
1979	Em 9 projectos, um valor de investimento de $3.2M USD nunca foi completado, $2M USD nunca foi utilizado, $1.3M USD foi abandonado, $0.2M USD foram utilizados com algumas alterações e apenas $0.1M USD foi utilizado como entregue [General79].
1982	75% dos sistemas desenvolvidos nunca foram completados ou utilizados [Gladden72].
1984	Em 2000 empresas, 40% dos sistemas falharam o atingimento dos resultados esperados [Bikson84].
1987	75% dos sistemas de controle da produção e inventário implementados tiveram problemas [Works87].

1988	Num universo de 34 analistas de sistemas, 70% consideraram que entre 20% e 50% dos projectos falham, porque não são satisfeitos os requisitos de negócio previstos [Lyytinen].
1994	Em 82 executivos entrevistados, 22% tinham abandonado mais de 5 projectos nos 5 anos anteriores e 69% abandonaram pelo menos 1 [Ewusi-Mensah].
1995	Em 143 projectos, 25% não respondiam aos requisitos [Phan95].
1995	Num universo de 365 empresas, 31% projectos cancelados antes do fim, 53% ultrapassaram custos; só 12% de 3682 foram completados a tempo e nos custos previstos [Johnson95].

Tabela 1.3: Estatísticas diversas obtidas ao longo do tempo sobre os projectos de desenvolvimento de software.

Foi precisamente este tipo de problemas que motivou a designação de "crise no software" já durante a década de 70, a qual foi reforçada por Fred Brooks no seu célebre artigo "*No Silver Bullet*" ("não existem balas de prata") [Brooks86], no qual este refere que dificilmente se encontraria uma cura milagrosa que pudesse resolver os problemas associados ao processo de desenvolvimento de software.

Os problemas até agora referidos têm muito a ver com questões que se verificam durante o processo de desenvolvimento de software, mas igualmente graves são as situações que podem ocorrer depois deste processo estar concluído, e os sistemas entrarem em produção. Neste caso, o adequado funcionamento dos sistemas é crucial para a existência e sobrevivência das organizações e das pessoas envolvidas, a diferentes níveis, envolvendo questões económicas, de segurança, privacidade, qualidade de vida, etc. Existem diversos casos clássicos que apontam para as falhas do software em funcionamento:

- Em 1979, ainda durante o período da guerra-fria, o mundo pode ter estado à beira de uma guerra nuclear quando o sistema americano que controlava o espaço aéreo detectou o lançamento de mísseis pela União Soviética em direcção aos Estados Unidos; de facto, tratava-se de um ataque simulado, e apesar de não terem sido divulgados muitos detalhes, parece legítimo supor que tal se tratou de um erro do sistema [Neumann80].

- Durante a guerra do Golfo, uma falha no software dos mísseis Patriot que os Estados Unidos enviaram para a zona da guerra não foi atempadamente detectada, e a correcção só chegou um dia após um ataque iraquiano com mísseis ter causado a morte a cerca de trinta soldados americanos [Mellor94].
- Devido a um erro no software de controlo de um equipamento médico, pelo menos dois doentes morreram entre 1985 e 1987 em consequência de terem recebido doses exageradas de radiação [Leveson93].
- Problemas diversos no software de controlo da distribuição e encaminhamento de bagagem do aeroporto de Denver, nos Estados Unidos, provocaram custos superiores a 1 milhão USD por dia [Gibbs94].
- Em 1995, estimativas diversas apontavam para que o custo dos projectos de software que foram abandonados nos Estados Unidos equivaleria a cerca de 80 mil milhões USD, qualquer coisa como 1% do PIB americano [Johnson95].

Como se pode constatar dos exemplos anteriores, os problemas resultantes do mau funcionamento ou do processo de desenvolvimento de software podem ter impacto em duas áreas críticas: questões financeiras por um lado, e vidas humanas por outro. Mesmo após a entrada em funcionamento do software, poder-se-ia pensar que o número de problemas iria diminuir drasticamente e estabilizar num nível muito baixo, idealmente próximo de 0. Tal não acontece, o que tem muito a ver com o facto de qualquer intervenção posterior à implementação do software poder vir a gerar um conjunto de problemas não previstos, e consequentemente um acréscimo de erros.

Diversas causas estão na origem deste crónico falhanço dos projectos de sistemas de informação, nomeadamente:

- Falta de empenhamento dos órgãos de topo das organizações.
- Falta de comprometimento e empenhamento dos utilizadores.
- Incompreensão do valor dos sistemas de informação.
- Falta de entendimento e de sintonia entre informáticos e clientes utilizadores do sistema, no âmbito e requisitos do mesmo.
- Deficiências várias no processo de desenvolvimento.

- Falhas na coordenação do projecto, nomeadamente ao nível da definição dos objectivos e das prioridades e da elaboração de estimativas.
- Falta de qualidade e inadequação dos recursos envolvidos.
- Mudanças frequentes dos requisitos do negócio e incapacidade de lidar com esta situação.
- Dificuldades na integração de componentes.
- Qualidade e desempenho do software deficiente, muito relacionados com problemas ao nível do controle de qualidade.
- Incapacidade de identificar e controlar os riscos do projecto.

1.8 Planeamento Estratégico de Sistemas de Informação

No passado, os sistemas de informação foram desenvolvidos simplesmente para melhorar a eficiência de determinadas funções de negócio; mais recentemente passaram a concentrar-se na obtenção de vantagens competitivas. Este facto justifica a inclusão de considerações sobre as tecnologias de informação na definição de estratégias do negócio, tal como é defendido por McFarlan [McFarlan83].

Um dos objectivos dos sistemas de informação é a satisfação adequada dos requisitos de negócio, garantindo assim o correcto alinhamento com a estratégia da organização. É por isso importante que, antes de se iniciar qualquer processo de desenvolvimento de componentes da arquitectura de sistemas de informação, que a mesma seja pensada de um ponto de vista global, garantindo assim a completa integração entre os componentes e a prioritização da respectiva implementação.

É esse o âmbito do Planeamento Estratégico de Sistemas de Informação, cujo principal resultado é um Plano Estratégico de Sistemas de Informação (ou Plano Director de Sistemas), que define os componentes do sistema de informação a implementar, e funciona como um guia para todas as futuras intervenções na área de informática. Na sequência deste plano, devem ser identificadas e prioritizadas as acções a desencadear para atingir a situação futura proposta. Só depois se entra no âmbito do desenvolvimento dos sistemas de informação, naquilo que normalmente se designa por Engenharia de Software. Por

esta razão, é também frequente que a actividade de planeamento estratégico seja designada por Engenharia de Sistemas, para traduzir a ideia de uma perspectiva mais abrangente.

Este tipo de abordagem pode ser aplicado numa organização já existente, sendo nesse caso necessário identificar o diferencial entre a situação actual dos sistemas de informação e o conjunto de recomendações elaborado.

Conceito

Podemos definir o **Planeamento Estratégico de Sistemas de Informação** (PESI) como um processo cuja finalidade é garantir o alinhamento dos sistemas de informação com os objectivos do negócio ou como Lederer referiu [Lederer88] "o PESI é o processo de decidir os objectivos para a organização informática e identificar as aplicações informáticas potenciais que a organização deve implementar".

Enquanto processo tem uma sequência de fases, cada uma com actividades e objectivos bem identificados (ver Figura 1.4).

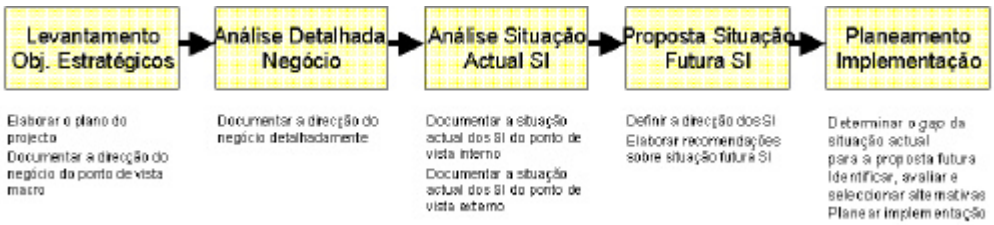


Figura 1.4: Metodologia de Planeamento Estratégico de Sistemas de Informação.

O objectivo deste processo é realizar um conjunto de actividades de levantamento de informação, do negócio e dos sistemas de informação, durante as três primeiras fases, de modo a que na quarta fase de possam elaborar recomendações sustentadas e que possibilita a elaboração de planos do projecto. No final do processo de PESI dispõe-se de um plano estratégico de sistemas de informação bem documentado, uma compreensão detalhada da situação actual do negócio e dos sistemas de informação e uma definição da direcção dos sistemas de informação suportada por toda a organização.

1.9 Engenharia de Software

Depois de definida uma estratégia global e identificados os componentes que é necessário desenvolver, a sua concretização passa para o domínio de outra "ciência", a Engenharia de Software. Esta inclui todas as actividades que vão desde um planeamento inicial do projecto até à instalação do sistema em produção, e posterior suporte. Por isso, disciplinas como análise de sistemas, gestão de projectos, programação, controle de qualidade poderão ser incluídas no âmbito da Engenharia de Software, conforme aqui a entendemos.

Uma das primeiras definições de Engenharia do Software foi dada por Fritz Bauer, nos finais da década de 60, como sendo "a definição e utilização de princípios de engenharia sólidos, de modo a desenvolver software económico, fiável e que trabalha eficientemente em máquinas reais. Inclui pois um conjunto de métodos, de ferramentas e de procedimentos". No entanto, esta definição peca por não fazer qualquer referência a aspectos técnicos, não referir a importância da satisfação do cliente, do cumprimento de prazos, da utilização de métricas e não enfatizar a importância de se utilizar um processo maduro.

Uma das definições mais utilizada hoje em dia foi proposta pelo IEEE em 1993, e defende que "a **Engenharia de Software** é a aplicação de um processo sistemático, disciplinado, e quantificado ao desenvolvimento, operação e manutenção de software; ou seja, a **Engenharia de Software** é a aplicação de técnicas de engenharia ao software".

Conceito

As actividades associadas à Engenharia de Software podem ser agrupadas em três grandes fases, tendo em conta que o seu objectivo é o desenvolvimento e operação de um produto: concepção, implementação e manutenção. Cada uma destas fases pode ainda ser dividida em outras mais elementares (ver Capítulo 2 para mais detalhes). Ao longo de cada fase existem tarefas, subprodutos a desenvolver, pontos de verificação e intervenientes. Existe também um conjunto de actividades de suporte contínuas: gestão de projecto, controle de qualidade, gestão da configuração, elaboração de documentação, elaboração de estimativas, gestão do risco, entre outras. É pois uma área de conhecimento

muito vasta, o que torna ainda mais difícil a sua aplicação de forma rigorosa e sistemática.

Apesar de se reconhecer o valor de um planeamento global com uma perspectiva integradora, muitas organizações estão mais preocupadas em resolver problemas imediatos e parciais, e por isso desencadeiam projectos individuais, que muitas vezes contemplam apenas as necessidades e requisitos de uma área restrita da organização. Assim, o esforço da implementação de sistemas de informação começa logo pelas actividades que já se situam no domínio da engenharia de software.

A Figura 1.5 ilustra de uma forma esquemática a discussão anterior entre as grandes áreas de Planeamento Estratégico de Sistemas de Informação e de Engenharia de Software, e suas relações. Através dela podemos perceber que, por exemplo, a Engenharia de Software inclui diversas questões que pertencem à Gestão de Projectos (planeamento, execução e aompanhamento de um projecto), mas está fora do seu âmbito questões como gestão de recursos humanos. A figura ilustra complementarmente o contexto e o foco primordial deste livro, designadamente os assuntos conotados com as abordagens orientadas por objectos, e em particular o UML.

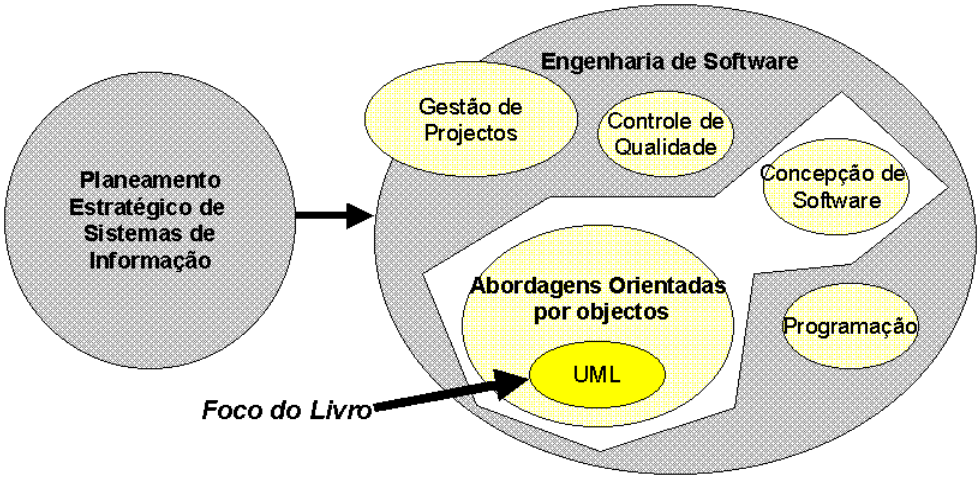


Figura 1.5: Relação entre PESI e Engenharia de Software, e o foco do livro.

Existem diversos produtos construídos pelo homem que apresentam problemas, mas mais raramente que os que se verificam no software. Por exemplo, um frigorífico pode falhar, mas menos do que um programa de contabilidade; uma ponte pode cair, mas tal é com certeza menos frequente do que a ocorrência de problemas nos sistemas operativos dos computadores. A ideia de que a concepção, implementação e manutenção do software poderiam ser realizadas aplicando técnicas tradicionais de engenharia levou a que já em 1967 um grupo de trabalho da NATO propusesse pela primeira vez o termo Engenharia de Software, e que este assunto fosse discutido em larga escala durante a conferência *NATO Software Engineering Conference* realizada na 1968.

A conclusão que resultou desta reunião foi que se deveriam utilizar os princípios e paradigmas de outras disciplinas de engenharia já bem estabelecidas de modo a resolver o que na altura se designou por crise do software (a qualidade do software era inaceitavelmente baixa e os custos e prazos não eram cumpridos). Actualmente há quem considere que a expressão mais adequada não seria “crise” mas sim “depressão”, dada a duração que ela já tem e ao facto de não se vislumbrar uma solução imediata.

1.10 Conclusão

Este capítulo de introdução tem como objectivo dar uma ideia dos principais problemas e preocupações que de um ponto de vista genérico se colocam aos intervenientes no processo de gestão e desenvolvimento informático. Estes problemas têm sido uma constante desde o início do desenvolvimento de software, e independentemente das iniciativas desenvolvidas, não foi possível até à data eliminá-los integralmente.

Devido a esta falta de resultados, poderíamos considerar que estávamos perante um facto consumado e inevitável, e abandonar os esforços no sentido de corrigir as falhas detectadas. A mensagem dos restantes capítulos, e do livro no seu todo, é a de que a nossa postura não pode nem deve ser esta, e que devemos continuar a procurar ultrapassar os problemas existentes, tal como os cientistas em medicina não desistem de procurar a cura para o cancro, apesar de já o tentarem há muitos

anos. Devemos recolher os exemplos das histórias de sucesso e daquilo que se consideram as melhores práticas de desenvolvimento de software. Nos próximos dois capítulos iremos abordar estas questões de um ponto de vista evolutivo e mais abrangente.

Tivemos também como objectivo a introdução e enquadramento sucinto de alguns conceitos desta área de engenharia, nomeadamente os comceitos de sistemas de informação, arquitecturas de sistemas de informação, planeamento estratégico de sistemas de informação, e engenharia de software. Esta discussão de conceitos permitiu ainda definir e explicitar claramente o âmbito e o contexto deste livro.

1.11 Exercícios

Ex.1. Explique, com base na definição de Anthony, a diferença entre um sistema de informação operacional, táctico e estratégico. Dê exemplos para clarificar a sua justificação.

Ex.2. Discuta a noção de sistema de informação face à noção de software. Com base na definição dada no livro pode-se ter um sistema de informação sem software? Justifique.

Ex.3. Enumere os três principais problemas relacionados com o desenvolvimento de sistemas de informação. Enumere três das causas conhecidas.

Ex.4. Justifique a importância do PESI (planeamento estratégico de sistemas de informação).

Ex.5. A flexibilidade é frequentemente referida como um dos atributos relacionados com a existência de qualidade num sistema de informação. Discuta os aspectos positivos e identifique possíveis consequências negativas.

Ex.6. De um modo geral, as opiniões expressas pelos executivos do negócio sobre os informáticos são criticas face ao seu conhecimento do negócio, mas não tecem grandes considerações relativamente a questões de natureza técnica. Indique algumas razões por que tal acontece.

Ex.7. Na sua opinião, existem situações em que faz sentido ter um processo conjunto de actividades de planeamento estratégico de sistemas de informação e de engenharia de software? Justifique a sua resposta.