A Orientação para Objectos: Problemática da Gestão de Custos nos Sistemas de Informação

Maria Clara Silveira
Rute Maria Abreu

Resumo:

A análise e concepção Orientadas para Objectos (OO) são métodos para a construção de sistemas. O seu conceito básico é que os sistemas de informação deveriam ser modelados em compilações de objectos individuais, tratando-os como exemplos de uma classe no interior de uma hierarquia de classes. Ao empregarmos a concepção OO estamos a criar um software que se adapta à mudança e à escrita com economia de expressão e com a obtenção de um nível de confiança muito maior em relação à exactidão do nosso sistema de software. Finalmente, espera-se reduzir custos inerentes ao desenvolvimento dos sistemas de software complexos. Em termos de custos, os produtos de software contêm mais complexidade que os outros. Booch (1994) afirma que a complexidade lhe é inerente e justifica indicando quatro factores: a complexidade do domínio do problema, a dificuldade em gerir o processo de desenvolvimento, a flexibilidade exigida e os problemas que caracterizam o comportamento dos sistemas discretos. Apresenta-se um aproximação da gestão de custos que permitirá à empresa competir e manter uma posição concorrente no mercado. Assim, uma filosofia para obter esta redução será implementar um SI com uma aproximação OO e arquitectura Internet. Inicialmente, a implementação destes sistemas poderão aumentar os custos, mas que serão compensados numa perspectiva de investimento de médio/longo prazo. A aproximação OO promove a reutilização, a continuidade e a compatibilidade. Esta última, é a integração do software noutros sistemas e é concretizada promovendo a homogeneidade na concepção e o estabelecimento de normas; a continuidade é a adaptabilidade a modificações nas especificações e é alcançada pelo desenvolvimento da simplicidade e a descentralização; a reutilização é a possibilidade de utilizar software noutras aplicações.

Área temática: *GESTÃO DE CUSTOS E SISTEMAS DE INFORMAÇÃO*

A ORIENTAÇÃO PARA OBJECTOS: PROBLEMÁTICA DA GESTÃO DE CUSTOS NOS SISTEMAS DE INFORMAÇÃO

SILVEIRA, Maria Clara
Mestre em Engenharia Electrotécnica e de Computadores - Informática Industrial (Faculdade de Engenharia - UPorto)
Início dos estudos de doutoramento na FE -UPorto
Professora Adjunta, nomeação definitiva, do Departamento de Informática
Escola Superior de Tecnologia e Gestão- Instituto Politécnico da Guarda
e-mail: silveira_ipg@hotmail.com
ABREU, Rute Maria
Mestre Engenharia Industrial (FCT - UNLisboa)
Doutoranda "*Nuevas Tendencias en Dirección de Empresas*" na Universidade de Salamanca - Espanha
Professora Adjunta, nomeação definitiva, do Departamento de Contabilidade e Auditoria
Escola Superior de Tecnologia e Gestão- Instituto Politécnico da Guarda
e-mail: ra_ipg@hotmail.com
Trav. do Ferreiro, 2 – 6200-356 Covilhã - Portugal
Telef. + 351 275 330 800 Fax + 351 275 330 809

Área Temática: (05) – GESTÃO DE CUSTOS E SISTEMAS DE INFORMAÇÃO

KEYWORDS: Gestão de Custos, Sistemas de Informação, Orientação por Objectos, Contabilidade dos Sistemas de Informação

A ORIENTAÇÃO PARA OBJECTOS: PROBLEMÁTICA DA GESTÃO DE CUSTOS NOS SISTEMAS DE INFORMAÇÃO

Área Temática: 5 – GESTÃO DE CUSTOS E SISTEMAS DE INFORMAÇÃO

ABSTRACT:

A análise e concepção Orientadas para Objectos (OO) são métodos para a construção de sistemas. O seu conceito básico é que os sistemas de informação deveriam ser modelados em compilações de objectos individuais, tratando-os como exemplos de uma classe no interior de uma hierarquia de classes. Ao empregarmos a concepção OO estamos a criar um *software* que se adapta à mudança e à escrita com economia de expressão e com a obtenção de um nível de confiança muito maior em relação à exactidão do nosso sistema de *software*. Finalmente, espera-se reduzir custos inerentes ao desenvolvimento dos sistemas de *software* complexos.

Em termos de custos, os produtos de *software* contêm mais complexidade que os outros. *Booch* (1994) afirma que a complexidade lhe é inerente e justifica indicando quatro factores: a complexidade do domínio do problema, a dificuldade em gerir o processo de desenvolvimento, a flexibilidade exigida e os problemas que caracterizam o comportamento dos sistemas discretos.

Apresenta-se um aproximação da gestão de custos que permitirá à empresa competir e manter uma posição concorrente no mercado. Assim, uma filosofia para obter esta redução será implementar um SI com uma aproximação OO e arquitectura *Internet*. Inicialmente, a implementação destes sistemas poderão aumentar os custos, mas que serão compensados numa perspectiva de investimento de médio/longo prazo.

A aproximação OO promove a reutilização, a continuidade e a compatibilidade. Esta última, é a integração do *software* noutros sistemas e é concretizada promovendo a homogeneidade na concepção e o estabelecimento de normas; a continuidade é a adaptabilidade a modificações nas especificações e é alcançada pelo desenvolvimento da simplicidade e a descentralização; a reutilização é a possibilidade de utilizar *software* noutras aplicações.

1. Introdução

Durante muitos anos o processo de desenvolvimento de *software* nas empresas resumia-se praticamente à programação e testes do *software*. Actualmente com o desenvolvimento das metodologias, nomeadamente: OO, *Internet*, entre outros, o grande desafio das organizações tem sido controlar os custos.

Apresenta-se um aproximação da gestão de custos que permitirá à empresa competir e manter uma posição concorrente no mercado. Assim, uma filosofia para obter esta redução será implementar um SI com uma aproximação OO e arquitectura *Internet*. Inicialmente, a implementação destes sistemas poderão aumentar os custos, mas que serão compensados numa perspectiva de investimento de médio/longo prazo.

A aproximação OO promove a reutilização, a continuidade e a compatibilidade. Esta última, é a integração do *software* noutros sistemas e é concretizada promovendo a homogeneidade na concepção e o estabelecimento de normas; a continuidade é a adaptabilidade a modificações nas especificações e é alcançada pelo desenvolvimento da simplicidade e a descentralização; a reutilização é a possibilidade de utilizar *software* noutras aplicações.

Assim, um SI desenvolvido numa perspectiva OO fica sensível ao contexto do produto e do utilizador. Sem a intervenção do engenheiro de sistema, o utilizador poderá alterar a estrutura dos dados, introduzindo novos campos na base de dados. Além disso os componentes do sistema tendem a ter uma estrutura fluída e não rigidamente hierarquizada.

O presente artigo desenvolve-se por um primeiro capítulo de introdução à problemática da gestão de custos na análise e concepção OO, que são métodos para a construção de sistemas de informação, implicando uma gestão eficaz e eficiente dos mesmos. No segundo capítulo procura-se centrar a temática da análise e concepção OO. Posteriormente, no terceiro capítulo, desenvolve-se a temática dos custos nas decisões que concernem aos SI. Finalmente, são tecidas as considerações finais e futuros desenvolvimentos sobre o trabalho apresentado.

2. Análise e Concepção Orientada para Objectos

A prática de cada disciplina de engenharia, quer seja civil, mecânica, química, electrónica ou de *software*, envolve elementos tanto da ciência como da arte. A função de um engenheiro de *software* enquanto artista é um autêntico desafio, quando a tarefa consiste em conceber um SI completamente novo ou "remendar" um sistema em funcionamento.

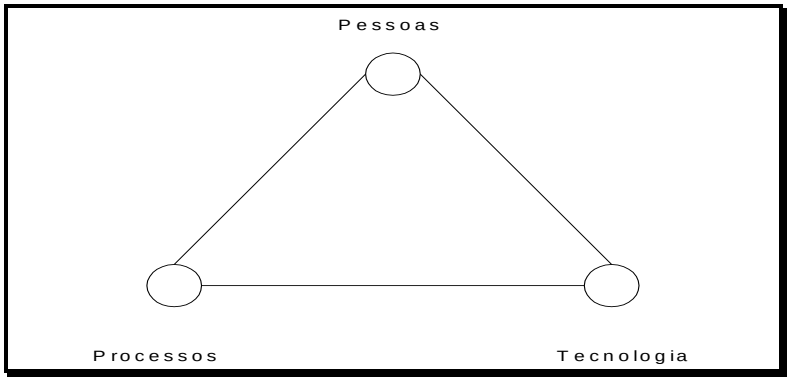
A análise e concepção encerram a abordagem disciplinada que se utiliza na procura de uma solução para certos problemas, providenciando, assim, uma ponte entre as necessidades e a implementação e no âmbito da engenharia de *software*, o seu objectivo é:

- construir um sistema que corresponda às especificações;
- que se adapte às limitações do meio;
- satisfazer as necessidades do cliente e as restrições no próprio processo de concepção, tais como a extensão, as ferramentas disponíveis e o custo.

Assim, a concepção envolve o equilíbrio entre um conjunto de necessidades competitivas. Os produtos da concepção são modelos que permitem raciocinar sobre as estruturas, realizar trocas sempre que os requisitos entram em conflito e, em geral, fornecer uma cópia do projecto para a implementação.

O processo de desenvolvimento de um sistema de *software* pode ser definido como um conjunto de actividades e factores envolvidos na sua produção. Estas actividades e factores estão relacionados entre si num processo coerente e controlado expresso na figura 1.

Figura 1 - Factores chave no processo de desenvolvimento



Fonte: *Piattini et al.*, 1995

O suporte qualitativo e produtivo do processo de desenvolvimento tem-se baseado essencialmente na introdução de novas tecnologias e em recursos humanos cada vez mais especializados e com maiores níveis de formação científica e técnica. Porém, na maioria das organizações, não se tem obtido os níveis de produtividade e qualidade desejados. Contudo, *Piattini et al.* (1995) defende a necessidade de estimular o esforço constante em construir infra-estruturas para o processo de desenvolvimento adoptando técnicas rigorosas de engenharia de *software* e uma adequada gestão de projectos.

Este artigo pretende alertar e envolver os utilizadores no decorrer do processo de desenvolvimento, já que a presença deles relembra constantemente o porquê e para quem o *software* está a ser desenvolvido**.** Neste sentido, a aproximação OO promove mecanismos nos quais os utilizadores e os analistas têm que cooperar.

O maior problema com os modelos de dados clássicos, tal como o modelo relacional, é que eles mantêm uma orientação fundamentalmente de registo. Por outras palavras, o significado da informação na base de dados - a sua semântica - não é facilmente notória a partir da própria base de dados para os utilizadores finais. Por esta razão, tem sido proposto um certo número de modelos semânticos de dados (SDM´s). Estes tentam fornecer meios mais expressivos para representar o significado da informação. Provavelmente, o SDM mais frequentemente citado é o modelo entidade relacionamento (modelo E-R). Neste modelo, o mundo real é representado em termos de: entidades, as suas relações e os atributos que lhe estão associados. Assim, os SDM´s contribuíram para o desenvolvimento da OO, como um:

- meio de clarificar a arquitectura de sistemas de gestão de bases de dados avançados;
- e uma disciplina para os sistemas de análise e concepção de bases de dados.

A modelização de entidades têm muito em comum com a modelização de objectos. A grande diferença que existe entre entidades e objectos reside no modo como estas construções são aplicadas no processo de modelização. As entidades são primariamente construções estáticas. O modelo E-R dá um suporte útil para descrever o pormenor estrutural de um sistema de bases de dados.

A metodologia proposta por *Booch* (1994) para o processo de desenvolvimento OO (micro), considerando que o cliente/utilizador o deve sempre validar, é composta pelas seguintes actividades :

- Identificar para um determinado nível de abstracção

- Identificar a semântica
- Identificar as relações os objectos e as classes
- Implementar

A primeira actividade, identificar para um determinado nível de abstracção os objectos e as classes, tem como objectivo estabelecer as fronteiras do problema em mão e assim subdivide-se:

- Na análise descobre aquelas abstracções que forma o vocabulário do domínio do problema e assim, começa a limitação do problema, decidindo aquilo que é importante, do que o não é;
- Na concepção dá-se o passo a fim de se inventar novas abstracções que formam elementos da solução.

A segunda actividade do processo de desenvolvimento micro é identificar a semântica de classes e de objectos, cujo objectivo é estabelecer o comportamento e os atributos de cada abstracção identificada na fase anterior. Aqui, aperfeiçoa-se as abstracções candidatas através de uma distribuição mensurável e inteligente das responsabilidades.

A identificação das relações entre classes e objectos é a terceira actividade e tem como objectivo solidificar as fronteiras e reconhecer os colaboradores com cada uma das abstracções identificadas desde o início do processo. Esta actividade formaliza as separações físicas e conceptuais que causem problemas entre as abstracções iniciadas na fase anterior, logo:

- Na análise aplica-se esta tarefa para especificar as associações entre classes e objectos (incluindo certas heranças de interesse e relações no conjunto), exprimir a existência de uma associação identificando alguma independência semântica entre duas abstracções, assim como alguma habilidade em passar de uma entidade para outra;
- Na concepção é para especificar as colaborações que formam os mecanismos da nossa arquitectura, assim como o agrupamento, a mais alto nível, de classes em categorias e de módulos em subsistemas.

A última actividade é a implementação de classes e de objectos. Durante esta fase, pode-se actualizar o dicionário de dados, incluindo as classes e os objectos novos, descobertos e inventados durante a formulação da implementação das abstracções existentes. Criam-se representações tangíveis das nossas abstracções a fim de apoiarem a afinação contínua das versões executáveis no processo de desenvolvimento iterativo e incremental.

Os projectos orientados ao objecto bem sucedidos apresentam:

- interactividade, no sentido em que envolve a afinação contínua de uma arquitectura OO, a partir da qual se aplica a experiência e os resultados de análise e concepção anteriores.
- incrementalidade, no sentido em que cada passo através de um ciclo de análise *vs.* concepção *vs.* evolução implica o aperfeiçoamento gradual e acaba por convergir para uma solução que está de acordo com as últimas exigências reais do utilizador e, além disso, é simples, fiável e adaptável.

As tecnologias OO reflectem uma visão natural do mundo. Os objectos são classificados em classes e estas são hierarquizadas. Cada classe contém um conjunto de atributos que a descrevem e um conjunto de operações que definem o seu comportamento. Alguns conceitos importantes em OO são a abstracção, encapsulamento, hereditariedade e polimorfismo.

No desenvolvimento de sistemas de *software* surgem inevitavelmente problemas mais ou menos complexos. O grau de complexidade relaciona-se com o domínio do problema e entre outros factores com a capacidade intelectual do ser humano.

Segundo *Coad et al.*(1991), citando as experiências de *Miller*, uma pessoa pode abranger apenas cerca de sete (± duas) fracções de informação de uma só vez na memória de curta duração. O alcance da memória imediata impõe portanto severas limitações na quantidade de informação que somos capazes de receber, processar e recordar. Existem factores limitativos fundamentais do conhecimento humano, estes factores podem ser tratados através do uso da decomposição, abstracção e hierarquia. Assim, a resolução de problemas passa pela divisão em fracções, processo que actualmente se designa por abstracção e quando se usa, admite-se que se está a considerar que é complexo e em vez de se tentar compreender o todo, selecciona-se parte dele. Sabe-se que existem detalhes adicionais, simplesmente opta-se por não os considerar naquele momento. Esta técnica é uma forma importante de gerir a complexidade.

Os modelos de análise e concepção OO reflectem a importância que tem a apreensão explícita das hierarquias de classe e de objecto do sistema em construção. A problemática da complexidade no domínio do problema segundo *Booch* (1994) resulta expressa em quatro atributos comuns a todos os sistemas, isto é:

- complexidade toma a forma de uma hierarquia, pelo que, um sistema complexo é composto por subsistemas inter-relacionados.
- natureza dos componentes primitivos de um sistema complexo é relativamente arbitrária e depende bastante do discernimento do observador do sistema. Aquilo que é primitivo para um observador, pode ser de um mais alto nível de abstracção para outro.
- ligações intra-componentes são, geralmente, mais fortes que as ligações inter-componentes. A diferença entre estas interacções providencia uma clara separação das relações entre as várias partes do sistema, tornando possível o estudo de cada parte de uma forma relativamente isolada.
- sistemas hierárquicos são, geralmente, compostos por apenas alguns tipos de subsistemas em várias combinações e disposições. Assim, os sistemas complexos possuem várias configurações comuns. Estas configurações podem incluir a reutilização de pequenos componentes.
- funcionamento de um sistema complexo, implica que se desenvolveu a partir de um sistema simples que funcionou. O procedimento de funcionamento implica sempre com um sistema simples que funcione, porque a partir do nada nunca funciona, nem existe maneira de o pôr a funcionar, há que recomeçar tudo de novo.

À medida que os sistemas se desenvolvem, os objectos que tinham sido considerados complexos tornam-se objectos primitivos. Para modelar estes objectos primitivos primeiro, temos de os utilizar num contexto simplificado depois, melhorá-los à medida que aprendemos mais sobre o comportamento do sistema.
Ao empregarmos a aproximação OO estamos a criar um *software* que se adapta à mudança e à escrita com economia de expressão e com a obtenção de um nível de confiança muito maior em relação à exactidão do nosso sistema. Espera-se, assim reduzir custos inerentes ao desenvolvimento dos sistemas de *software* complexos.

As ferramentas *Computer Aided Software Engineering* (CASE) ao fornecerem apoio automatizado a qualquer notação, podem ajudar o engenheiro de sistemas a desenvolver

estimativas do custo do SI, adaptados a cada organização. Assim, para *Sommerville* (1996) a classificação funcional das ferramentas CASE encontra-se expressa no quadro 1.

Quadro 1 - Classificação funcional de ferramentas CASE

Tipo de ferramenta	**Exemplos de Aplicação**
Gestão projectos	PERT, ponderação de custos
Edição	Editores de texto, editores de diagramas
Gestão de configuração	Gestores de versões, gestores de mudanças
Prototipagem	Linguagens de alto nível, geradores de interface
Suporte a métodos	Editores de análise e concepção, dicionários de dados, geradores de código
Processamento (linguagens)	Compiladores, interpretadores
Análise de programas	Analisadores estáticos e dinâmicos
Teste	Geradores de dados de teste, comparação de ficheiros
Debugging	Sistemas de depuração interactivos
Documentação	*Layout* de programas, editores de imagem
Reengenharia	Sistemas de reestruturação

Fonte: *Sommerville* (1996)

Esta nova geração de ferramentas CASE já vem dotada de alguns meios para automatizar uma variedade de métricas de *software*. Incorporam também repositórios e/ou *Information Resource Dictionary* que têm vindo a adquirir uma maior importância na arquitectura destes ambientes, sendo até um elemento essencial para o desenvolvimento dos SI. Um repositório armazena toda a informação que diz respeito ao sistema: gráfica, dados, processos e regras de suporte das metodologias. Contém tudo o que é necessário para que o sistema possa evoluir e desenvolver-se à *posteriori*.

As ferramentas apenas dão maior poder ao engenheiro de sistema, libertando-o para que se possa concentrar nos aspectos realmente criativos da análise ou concepção. Assim, existem algumas tarefas em que as ferramentas são especialmente úteis, como por exemplo: especificação de requisitos, análise de custo/benefício, concepção, implementação, verificação de consistência, restrições e notação e, ainda para a validação. Contudo, em oposição, as ferramentas não dizem que se deve criar uma nova classe de forma a simplificar a estrutura de classes, isso requer a compreensão humana.

Poderíamos pensar em tentar utilizar um "sistema especialista" como ferramenta, mas isto exige uma pessoa especializada, tanto no desenvolvimento OO como no domínio do problema, e a capacidade de articular heurísticas de classificação, assim como um conhecimento extenso. Não esperamos que as ferramentas em questão surjam num futuro próximo, entretanto, temos sistemas reais para criar.

3. Problemática da Gestão de Custos

Segundo *Pires Caiado* (1997), a Contabilidade de Gestão deve apurar os custos de cada um dos segmentos (...) analisá-los e transmiti-los aos responsáveis, tendo em visto o seu controlo e adequação aos objectivos da empresa. Logo, cada organização estabelece

diferentes metas, fins ou objectivos, constituindo mesmo um desafio à criatividade de todos os agentes económicos, pelo que a sua concretização permite as empresas um desempenho excepcional. Mas, o reverso da "moeda" é a necessidade de planeamento, gestão, controlo e avaliação das suas actividades. A situação de "descuido" destas regras elementares da gestão ou administração das empresas conduz a casos bem conhecidos.

Por outro lado, *Peters* (1987), durante a avaliação de um inquérito que desenvolveu, refere a seguinte problemática "estávamos à espera de confirmar que as companhias por excelência colocassem grande ênfase nos aspectos dos custos, ou da tecnologia ou dos mercados (...), mas não estávamos a contar com nenhuma orientação particular. (...) Encontrámos um denominador comum importante: *tendem a ser movidas por atitudes que lhes permitam estar próximas dos clientes* do que por qualquer outra relacionada com tecnologia ou custos." Na realidade, a enorme difusão dos SI baseia-se precisamente nesta necessidade, que parece ser adoptada por grande número de empresas. Também no desenvolvimento do SI usando a aproximação OO implica a presença do cliente, pois relembra constantemente o porquê e para quem o sistema está a ser desenvolvido.

Para *Rodríguez* (1996) a empresa necessita de um SI para fazer factível a inter-relação do seu meio envolvente interno e externo e, assim aproveitar oportunidades e evitar perigos. A empresa ao possuir um SI considera, entre outras, as seguintes vertentes:

1. Utilidade na tomada de decisões;
2. Fontes de reserva da informação;
3. Canal de comunicação;
4. Sistema de processamento;
5. e a gestão de custos.

Para *Williamson et al.* (1975), o impacto da informação na gestão empresarial está estreitamente relacionado com as condições económicas, de tal forma que segundo afirmam o valor de esta aumenta quando existe incerteza. Os custos nas tecnologias de informação são cada vez maiores, mas o desenvolvimento de um sistema, implica a necessidade de realizar estimativas. Para concretizar tal, na óptica contabilistica e, especificamente, na Contabilidade de Gestão podemos encontrar diferentes conceitos/metodologias que nos permitem poder quantificar esta problemática e, então decidir.

O presente artigo pretende alertar para a problemática do custos do SI. Contudo, a preocupação de enumerar exaustivamente todos os custos subjacentes ao sistema e respectiva metodologia: directos, indirectos, básicos, reais, padronizados, fixos, variáveis, relevantes, irrelevantes, marginais, diferenciáveis, controláveis e não irrecuperáveis, tornaria este trabalho impossível de concretizar. Neste sentido, procura –se o maior detalhe possível, salvaguardando desde já a sua extensão.

Não se poderá deixar de referir que a classificação e/ou a padronização do custo no SI debate-se com algumas dificuldades de concretização, principalmente pelo elevado número de conceitos que utilizam a abstracção e artefactos. Neste sentido, o custo do SI pode ser considerado segundo a perspectiva do seu interveniente:

- T*otal Cost of Ownership* (TCO).

Este termo é bastante utilizado pelos engenheiros de sistemas e pretende calcular os custos (e benefícios) das organizações pelo facto de utilizarem tecnologias de informação. Esta

tarefa não é fácil, pois pode envolver custos tão divergentes como: equipamentos, passando pela instalação, configuração, formação, gestão, utilização, manutenção e monitorização dos sistemas e, ainda, todos os que, directa ou indirectamente, contribuem para a sua implementação como: processo de selecção, compra, pagamento e até mesmo de substituição. Contudo, esta classificação ou não ser tão alargada, limita o seu estudo.

- Custo do Sistema de Informação (CSI).

Este conceito é bastante usado pelos especialistas na área de contabilidade de gestão.
Em seguida, procurar-se-à desenvolver uma nova metodologia suportada numa proposta defendida por *Stiglitz* (1975) e *Rodríguez* (1996) em que CSI pode comportar a subdivisão em custos de:

1. Pesquisa;
2. Implementação;
3. Monitorização;
4. Imprevistos.

A primeira desagregação do CSI deve considerar nos custos de pesquisa, todos os que especificamente comportem as quatro fases consecutivas desta desagregação, ou seja:
1ª Fase - análise de alternativas no mercado dos SI,
2ª Fase – avaliação;
3ª Fase - selecção do SI adequado aos objectivos da empresa;
4ª Fase - e futura implementação no seio da empresa.

Nestas quatro fases anteriores podem ser envolvidos os seguintes custos de:

- Recursos Humanos, que podem ser engenheiros de sistemas e/ou da área do sistema a implementar, comportando: salários, encargos sociais, transportes, eventos, deslocações, as regalias previstas na lei (horas extra, subsidio de alimentação, natal e férias) e adicionais disponibilizadas pelas organizações (*plafond* de cartão de crédito, telemóvel, carro, entre outras).
- Bens/Serviços necessários ao desempenho das funções e tarefas, como: trabalhos especializados, equipamentos informáticos e administrativos, aluguer de espaços e material de escritório.
- Todos os custos que devem estar contidos nas funções de recolha, análise, selecção, comunicação, tratamento e apresentação de conclusões de projectos de investimento.
- Como envolve uma decisão, a empresa deveria incluir o custo de oportunidade, isto é, a decisão de implementar um dado SI, implica a rejeição de outros sistemas ou até mesmo não ter sistema nenhum.
- De facto, associado a esta 1ª fase deve também existir a análise de risco que resulta da ponderação da incerteza associada à decisão do SI a implementar.

Se por hipótese se recorrer a subcontratar uma empresa especializada na actividade de pesquisa e avaliação do SI, dever-se-á considerar o custo de: publicidade da necessidade do estudo, da selecção da empresa que vai desenvolver o projecto, do contrato e todos os inerentes a esta situação (contencioso, financeiros, extraordinários e até os imprevistos).

Não se pode deixar de referir a procura de estimativas nesta fase de pesquisa para os custos de implementação. Estes serão desenvolvidos e integrados nas três tarefas principais da análise e avaliação de projectos de investimento, isto é:
1. investimento propriamente dito, nas duas componentes de capital fixo (ex. *license cost*) e circulante (ex. fornecedores de equipamento).

2. financiamento com: negociação entre alternativas do mercado financeiro (ex. *leasing*) e de activos específicos (ex. locação operacional), encargos de negociação e o respectivo custo de oportunidade.
3. e a exploração previsional do SI a implementar:

i) Recursos Humanos, nomeadamente:

- Analistas, envolvendo entrevistas, preparação de relatórios, documentação, estudos, preparação de procedimentos, testes ao sistema, inspecções, revisões, formação, consultas a utilizadores, peritos, programadores, supervisão, desenho de documentos e apresentação formal;
- Programadores: codificação, depuração, testes, revisões, personalização de programas e consultas a analistas;
- Operadores: conversão, formação, suporte ao programador e consultas a analistas;
- Pessoal administrativo: conversão, formação e consultas a analistas;
- Direcção: supervisão e consultas a analistas.
- Especialistas na área ou no domínio do problema.
- Outros: segurança, entrevistas, preparação de relatórios, documentos, estudos, definição de procedimentos, tarefas administrativas, higiene e conforto.

ii) Bens/Serviços, ou seja: aquisição, instalação e adaptação de *hardware* e *software*, com testes de aceitação, utilização dos equipamentos existentes, conversão de ficheiros, testes ao sistema e ainda, consumíveis (publicação de manuais e procedimentos, papel, formulários, disquetes e CD) e política de seguros.

iii) Complementares, por exemplo: custos de interrupção de processos e procedimentos na empresa, suporte da direcção e administrativo e externos, entre outros: consultadorias, auditorias, formação especifica, de preparação do circuito de documentação, correcção monetária, diferenças cambiais, processos e procedimentos legais.

iv) Espaço, isto é, adaptação e/ou modificação de instalações para controlo de acessos e gabinetes, sistemas de infra-estrutura eléctrica, manutenção de ambiente (aquecimento, arrefecimento e níveis de humidade), de comunicação (sistemas, linhas e dados), segurança e vigilância (sistemas, técnicos e informação), higiene e conforto.

v) Inicialização do processo para todos os utilizadores do sistema que se também se podem desagregar em custos:

1. formação, adaptação, reconversão e actualização especificamente: formação especializada para desenvolvimento de uma área que pode ser nova ou apenas reciclagem, perda ou destruição (intencional ou não) de informação e sistemas, problemas psicológicos de aversão à mudança ou de dificuldades de adaptação, perda de produtividade nas funções que se encontrem afectadas pelas alterações;
2. bens/serviços que se considerem complementares, como: paneis/quadros de informação, novos equipamentos administrativos (secretarias e cadeiras ergonómicas), publicidade, entre outros;
3. outros custos, nomeadamente todos os que podem conduzir a uma melhor adaptação do SI na empresa, isto é, número e extensão de projectos face ao tamanho da empresa, alargamento dos tempos de paragem ou causas *downtime* planeadas (como: reparar, actualizar e adicionar novos componentes de *hardware* e *software*, *backups* e investimentos em componentes redundantes) apresentação e comunicação ao interior e exterior da empresa da implementação do SI.

Por outro lado, na segunda desagregação do CSI, deve-se considerar o custo de implementação. Este deve procurar corresponder às estimativas anteriormente referidas nos custos de previsão. Contudo, pode-se encontrar desvios (positivos ou negativos) face à realidade. Numa abordagem da NASA (1995) considera-se que o custo de implementação:

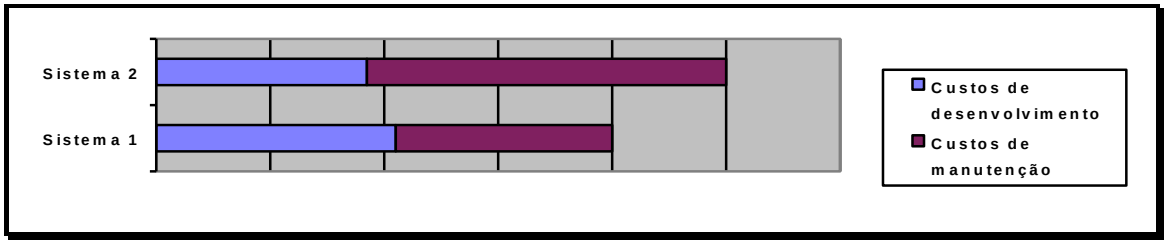
- não devem exceder 2% de custos de desenvolvimento ou manutenção do *software*;
- e a rubrica de recursos humanos afecta à componente técnica e de análise deve variar entre, respectivamente, 3% a 7% e 5% a 15% do total do valor estimado.

A terceira desagregação do CSI implica tratar o custo de monitorização que corresponde aos custos de operação e de manutenção do sistema em funcionamento contínuo. Neste âmbito, os custos envolvem:

- *hardware/software*, por exemplo: a gestão, manutenção e o tempo de utilização de sistemas, a memória principal/secundária, impressoras, redes, acessórios, dispositivos e operações de entrada e saída;
- recursos humanos, por exemplo: administração, engenheiros de sistema, programadores (manutenção), operadores, administrativos e seguranças;
- bens/serviços, ou seja, consumíveis como: papel, formulários, disquetes, inventariação, cópias e suporte, segurança e espaço (gabinetes), consultadoria e auditorias;
- imprevistos, todos os custos relativos a situações de risco ou condições de incerteza (extensão de dados, acumuladores de corrente por deficiências no fornecimento de energia eléctrica).

A figura 2 trata da questão do custo de manutenção *versus* o custo de desenvolvimento. Segundo *Sommmerville* (1996) observa-se que uma maior dedicação na tarefa desenvolvimento (e consequente aumento nos custos) provoca uma redução nos custos de manutenção, bem como no desempenho da mesma tarefa.

Figura 2: Custos de desenvolvimento e de manutenção



Fonte: *Sommmerville*,1996 (adaptação)

Complementarmente, no custo de monitorização engloba-se também o custo de qualidade. Este associa-se ao sistema de garantia da qualidade do SI e a situações-tipo: "fazer bem à primeira", "preventiva e evitar avarias", "reduzir custos inúteis", "gerir de forma optimizada" e o "princípio da excelência". Os custos que lhe estão associados comportam:

1. prevenção, este engloba processos de planeamento, revisões técnicas formais, constituição de equipas e respectiva formação;
2. avaliação, inclui as situações de inspecção e auditoria de processos, testes e cooperação em equipa dinâmicas e multidisciplinares;
3. deficiências no sistema, implicam os custos relativos a erros que se podem subdividir em internos e externos. Nos primeiros faz-se a revisão e reparação dos erros do sistema, enquanto os segundos resultam de queixas de utilizadores que implica diferentes formas de actuação (substituição de componentes e devolução do sistema, accionamento da garantia e suporte de linha de apoio)

Cita-se um estudo da NASA (1995), em que no custo de monitorização se pode também referenciar que a natureza dos erros de *software* incluem a frequência com que estes ocorrem, os custos de os localizar e remover, o nível de gravidade e as causas mais frequentes, assim como os processos efectivos de identificação e prevenção da sua

ocorrência. Com efeito, o impacto dos erros de *software* sobre os custos poderá ser muito elevado e na perspectiva de *Pressmann* (1997) "um erro que é detectado durante a concepção tem um custo de uma unidade monetária (u.m.)para a sua correcção. O mesmo erro detectado antes dos testes custa 6,5 u.m.; durante a fase de testes 15 u.m.; depois da entrega a correcção poderá custar entre 60 a 100 u.m.". O custo destas correcções cresce quase que em ordem geométrica, quando se percorre de forma ascendente as etapas do ciclo de vida (análise, concepção, implementação, teste e manutenção).

Na última desagregação do CSI, considera-se o custo associado a situações de imprevistos, principalmente medidos em tempo e valor. Este classifica-se como não planeados, respeitantes a erros, fraudes, omissões e falhas na selecção, na implementação, no desenvolvimento e monitorização de sistemas, alterações de mercado, modificações de variáveis macro-económicas, causas *downtime* não planeadas (por exemplo: terramotos, trovoadas, "cegonhas", situações de instabilidade, crises e greves) e o próprio ciclo de vida dos sistemas.

Na realidade, a ponderação dos custos é fundamental porque o investimento exigido representa para qualquer empresa um grande esforço. É assim que os métodos de cálculo destes custos são importantes, devendo ser realistas e exigindo-se técnicas especiais.

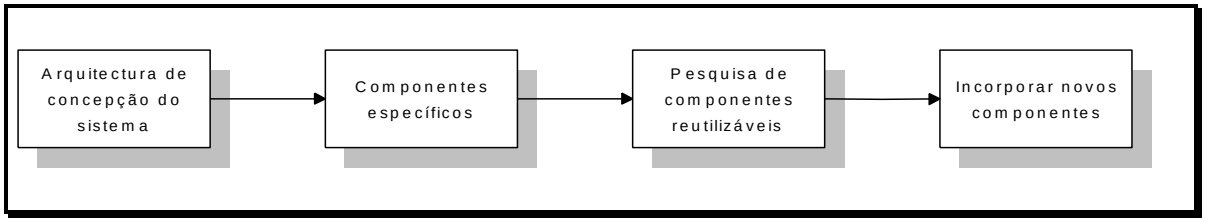
Segundo *Meyer* (1988), a reutilização, a continuidade e a compatibilidade são atingidas promovendo a modularidade do *software*, i.e., a decomposição dos sistemas num conjunto de módulos coerentes e as metodologias OO obedecem mais aos requisitos necessários para a obtenção de um sistema de *software* de qualidade do que a metodologia clássica.

Consegue-se reutilizar *software*, com a ajuda de ferramentas adequadas, podendo reutilizar todos os elementos que intervêm no desenvolvimento duma aplicação e esta etapa permite um decréscimo de custos, um aumento da produtividade e a consequente melhoria de qualidade.

Na perspectiva de *Pressman* (1997), os blocos construtivos de *software* estariam disponíveis para a construção de grandes programas com um menor esforço de desenvolvimento e devem ser catalogados para uma fácil referência, normalizados para uma fácil aplicação e validados para uma fácil integração.

Um produto de *software* que já existe pode constituir-se como protótipo para um novo produto eventualmente melhor e competitivo. De algum modo, esta é uma forma de reutilização na construção de protótipos expressa na figura 3.

Figura 3 - Paradigma reutilização de componentes de *software*



Fonte: *Sommerville*, 1996

Na presença de uma grande biblioteca com componentes de *software* reutilizáveis, o engenheiro de sistema deve reunir estas partes de forma inovadora para satisfazer as

necessidades estabelecidas e implícitas, assim como o pintor ou o músico devem ultrapassar os limites dos seus instrumentos. Infelizmente, visto que tais bibliotecas raramente existem para o referido engenheiro, geralmente, este terá de continuar o seu trabalho com um conjunto de facilidades relativamente primitivo (*Booch*, 1994).

A tipologia do CSI, apresentada anteriormente, é suportada pelo processo que lhe deu origem, mas se atendermos a metodologia da NASA (1995), especificamente as características do *software* então provavelmente o conjunto de atributos mais importantes incluem a produtividade, fases, actividades, alterações e todos os outros requeridos pelo planeamento, gestão e controlo do desenvolvimento de *software*, bem como a respectiva manutenção.

Os novos sistemas de informação suportados por técnicos de áreas inovadoras, sistemas sofisticados de *hardware* e *software*, novos processos de organização de estruturas funcionais e a implementação de circuitos de informação com objectivos de divulgação, salvaguarda da integridade do seu património, prevenção e detecção de erros, fraudes e omissões têm de implicar em custos que são ponderados na decisão de investimento.

4. Considerações Finais

Os Sistemas projectados segundo uma Orientação para Objectos tendem a exibir características bastante diferentes das que apresentam quando projectadas segundo as técnicas funcionais, implicando também a identificação de modelos e custos diferenciados.

Este artigo pretende alertar e envolver os utilizadores no decorrer do processo de desenvolvimento, já que a presença deles relembra constantemente o porquê e para quem o *software* está a ser desenvolvido**.** Neste sentido, a aproximação OO promove mecanismos nos quais os utilizadores e os analistas têm que cooperar.

A nossa experiência mostra que o desenvolvimento orientado ao objecto não é nem rigorosamente em "*top-down*", nem rigorosamente em "*bottom-down*". Em vez disso, os sistemas complexos bem estruturados são mais perfeitos se criados através do uso da concepção de forma cíclica. Este tipo de concepção dá ênfase ao desenvolvimento iterativo e incremental de um sistema através do aperfeiçoamento das diversas visões lógicas e físicas do sistema como um todo.

Por outro lado, as ferramentas CASE poderão auxiliar os gestores a medirem o desempenho do processo de desenvolvimento, constituindo-se como uma vantagem real pela diminuição dos custos que poderão representar. Os custos elevados na aquisição, formação e utilização destas ferramentas poderão ser minimizados se se tentar quantificar os custos da não adopção e a valorização dos benefícios que estas podem trazer como tecnologia estratégica, nomeadamente o aumento da produtividade e a redução dos custos.

Como principais benefícios dos SI segundo a abordagem OO podemos especificar:

- Obtenção de sistemas mais defendidos em relação a tentativas de corrupção quer propositadas, quer acidentais.
- Redução do custo total do ciclo de vida do SI, aumentando a produtividade dos engenheiros de sistema e reduzindo os custos.

- Maior compreensibilidade e facilidade de manutenção.
- Capacidade das tecnologias OO proporcionam mecanismos para modelizar a realidade com um grau de natural correspondência entre a realidade e o modelo, o que se aplica sobretudo em problemas onde as abordagens tradicionais falham, ou seja, sistemas concorrentes envolvendo processos em tempo real.

A aproximação de determinação da gestão de custos dos SI apresentada, ainda não foi testada. Neste sentido, a perspectiva futura será a aplicação e desenvolvimento à realidade empresarial, podendo concluir com os resultados a capacidade de implementar num futuro próximo um novo protótipo de *software* que agilize esta tarefa.

Outro desenvolvimento, não menos importante, será a medição da produtividade do desenvolvimento de *software* com vista à obtenção do controlo dos respectivos custos. Esta tarefa constitui uma dificuldade adicional que segundo *Banker (*1994) poderia ser superado pelas ferramentas CASE (especialmente: as integradas com repositório) que têm o potencial para apoiarem a automatização destas medições.

Apesar dos desenvolvimentos já referidos, também se pode avaliar o impacto e a utilização dos SI nas organizações, bem como outros relacionamentos entre contabilidade e as tecnologias de informação, constituindo apenas uma dificuldade o suporte em bases de dados, capazes de enquadrar estes estudos.

Bibliografia citada

- Banker, D et al. (1994), Automating Output Size and Reuse Metrics in a Repository-Based Computer-Aided *Software* Engineering (CASE) Environment; *IEEE*, Vol. 20, No.3, March, pp. 169-187.
- Booch, G. (1994), *Object-Oriented Analysis and Design with Applications*; The Benjamin/Cummings Publishing Company Inc, Redwood City, 2ª Ed.
- Coad, P. e Yourdon, E (1991), *Object-Oriented Analysis*; Prentice-Hall, 2ª Ed.
- Meyer, B. (1988), *Object Oriented Software Construction*, Prentice Hall.
- NASA (1995), *Software Engineering Program*, Software Measurement Guidebook, http://www.ivv.nasa.gov/.
- Peters, T. E Waterman, R. (1987), *In Search of Excelence*, Pub. Dom Quixote, Lisboa.
- Piattini, M., Sunil, G. e Daryanani, N. (1995), *Elementos y Herramientas en el Desarrollo de Sistemas de Información: Una visión actual de la tecnología CASE*; RA-MA Editorial, Madrid.
- Pires Caiado, A (1997), *Contabilidade de Gestão*, Vislis Editores, Lisboa.
- Pressman, R. (1997), *Software Engineering*, McGraw-Hill,. 4ª Ed.
- Rodriguez, S. (1996), *La relación de agencia en la empresa. Análisis y control de los costes de la agencia*. ICAC.
- Sommerville, I. (1996), *Software Engineering*, Addison-Wesley, 5ª Ed.
- Stiglitz, J. (1975), Incentives, Risk and Information: Notes towards a theory of hierarchy, *The Bell Journal of Economics*, vol. 6, pp. 552-579.

- *Williamson,O., Watcher, M. e Harris, J.* (1975),Understanding the employment relation: the analysis od idiosyncratic exchange, *The Bell Journal of Economics, Spring, pp. 250-278*