# PADRÕES: UMA FORMA DE REUTILIZAR CONHECIMENTO

Clara SILVEIRA Instituto Politécnico da Guarda

## RESUMEN

Aumentar la productividad y reducir los gastos por el hecho de reutilizar conocimiento durante el proceso de desarrollo de software es uno de los objetivos de las organizaciones actuales. Los padrones representan conocimiento que puede ser transmitido y enseñado. Un padrón es una solución para un problema general que puede ser aplicada en diferentes contextos. El estudio de padrones ocurre en distintas áreas del conocimiento humano. En realidad, la importancia de los padrones hace mucho que es reconocida en asignaturas como la biología, química, física y arquitectura [Booch, 1998]. También Simon [1975], en un estudio que desarrolló sobre la complejidad de los sistemas, ha observado la existencia de padrones.

Este trabajo se centra en el área de la ingeniaría de software y propone una abordaje para la reutilización de conocimiento a través de padrones. Este abordaje comprende técnicas de gestión del conocimiento y orientaciones para establecer un programa de reutilización del conocimiento.

**PALABRAS CLAVE:** Conocimiento, Padrones de software, Reutilización.

## RESUMO

Incrementar a produtividade e reduzir custos pelo facto de reutilizar conhecimento durante o processo de desenvolvimento de software é um dos objectivos das organizações actuais. Os padrões representam conhecimento que pode ser transmitido e ensinado. Um padrão é uma solução para um problema geral que pode ser aplicada em diferentes contextos. O estudo de padrões ocorre em várias áreas do conhecimento humano. Na realidade, a importância dos padrões há muito que é reconhecida em disciplinas como a biologia, a química, a física e a arquitectura [Booch, 1998]. Também Simon [1975], num estudo que desenvolveu sobre a complexidade dos sistemas, observou a existência de padrões.

Este trabalho enquadra-se na área da engenharia de software e propõe uma abordagem para a reutilização de conhecimento através de padrões. Esta abordagem compreende técnicas de gestão de conhecimento e orientações para estabelecer um programa de reutilização de conhecimento.

**PALAVRAS CHAVE:** Conhecimento, Padrões de software, Reutilização.

# 1. INTRODUÇÃO

O conceito de padrão (*pattern* em inglês) é muitas vezes referenciado pelos trabalhos de Alexander ([Alexander et al., 1977]; [Alexander, 1979]) que recolheu e documentou soluções genéricas para problemas recorrentes no domínio da arquitectura. Identificou 253 padrões relativos às descrições textuais de soluções para a criação de cidades, vilas, urbanizações, casas, edifícios e comunidades.

Com efeito, os padrões são úteis para ajudar as pessoas a reutilizar boas soluções para problemas recorrentes. Ajudam a comunicar melhor e a argumentar. Quando os padrões se relacionam formam uma linguagem a que se chama linguagem de padrões. A linguagem de padrões assegura a existência duma estrutura. Quando um grupo de pessoas trabalha com uma linguagem de padrões os pressupostos são explícitos desde o início o que trará benefícios para o sucesso [Alexander, 1979]. As linguagens de padrões não são linguagens formais, mas antes um conjunto de conhecimentos relacionados que fornecem um vocabulário comum sobre um problema particular.

A ideia de padrão foi extrapolada para o software, tendo sido desenvolvidos padrões para a modelação orientada por objectos ([Coad, 1992]; [Fowler, 1997]), para o desenho e programação orientada por objectos ([Coplien and Schmidt, 1995]; [Gamma et al., 1995]; [Vlissides et al., 1996]), para arquitecturas de software [Buschmann et al., 1996], para a exploração de técnicas, estratégias e aplicações [Rising, 1998] e para a escrita de requisitos de software ([Biddle et al., 2000]; [Adolph and Bramble, 2003]), entre outros.

Os padrões e as linguagens de padrões ajudam a equipa de desenvolvimento a comunicar conhecimento arquitectónico, a aprender um novo paradigma de desenho e ajudam os novos elementos da equipa a evitar erros que já foram detectados através de muita experiência entretanto adquirida [Schmidt et al., 1996]. Por outro lado, recolher e documentar a experiência, por exemplo na área de engenharia de requisitos, sob a forma de padrões, facilita a transferência de conhecimento para a prática [Hagge and Lappe, 2005].

[Gamma et al., 1995], descrevem soluções simples para problemas específicos do desenho de software orientado a objectos, isto é, recolhem soluções que foram desenvolvidas e aperfeiçoadas ao longo do tempo, apresentadas sob a forma de catálogo.

[Buschmann et al., 1996], indicam que um padrão de software descreve um problema recorrente num contexto específico e apresenta uma solução genérica para o resolver. A solução descreve os componentes que a constituem, as suas responsabilidades, as associações e os mecanismos de colaboração entre eles.

Porém, tal como refere Meyer [1997], um padrão não deve ser uma mera descrição em livro, mas sim um componente de software ou um conjunto de componentes. Também Booch [1998a] refere a identificação metódica de padrões com uma forma de reutilização a alto nível.

Um problema muito frequente relacionado com os padrões está no tempo exigido para a sua recolha e documentação. A "Lei de Pareto" também conhecida como regra 80/20, indica que 80 por cento do valor corresponde a 20 por cento de esforço ([Meszaros, 1996]; [Sparling, 2000]; [McConnell, 2004]). Com efeito, Jim Coplien mostrou que é possível obter conhecimento muito importante com muito menos esforço usando um formato de padrão mais compacto, ou seja, descreve-se cada padrão com a identificação do problema, contexto, forças que afectam o problema, solução e padrões relacionados [Meszaros, 1996].

## 2. REUTILIZAÇÃO DE CONHECIMENTO

As técnicas de gestão de conhecimento podem e devem ser aplicadas para promover a reutilização de conhecimento e melhorar os processos de desenvolvimento de software nas organizações.

A gestão do conhecimento refere-se à criação, identificação, integração, recuperação, partilha e utilização do conhecimento dentro da organização e pode ser entendida como a arte de gerar valor a partir de bens intangíveis da organização [Serrano e Fialho, 2003].

Para [Figueiredo, 2003], a designação de gestão do conhecimento, corresponde à forma de gerir os processos de construção social do conhecimento. A história da gestão dos conhecimentos, ao longo dos tempos, foi sempre construída em torno de quatro factores que estão intimamente ligados: domínios do conhecimento; actores (funções, papéis, profissões); formas de acção e / ou intervenção e modelos organizacionais. Ainda segundo o mesmo autor a gestão do conhecimento está ainda longe de se constituir como um corpo de práticas comprovadas.

Sempre que o equilíbrio entre estes factores foi perturbado, geraram-se situações de crise, que exigiram a procura de novos equilíbrios. Os novos equilíbrios foram sempre encontrados gerando outros domínios do conhecimento. Em consequência surgiam novas formas de acção e novos modelos organizacionais, com novos actores, novas profissões ou novas funções.

| **Taxonomia do Conhecimento** |
|---|
| **Conhecimento Computorizado** |
| Conhecimento Explicito |
| Conhecimento cuja semântica é conhecida pelo software |
| Conhecimento cuja semântica não é conhecida pelo software |
| Conhecimento Implícito |
| Conhecimento descoberto automaticamente |
| Conhecimento embebido nos processos de negócio das aplicações |
| **Conhecimento Não Computorizado** |
| Conhecimento gravado |
| Conhecimento não gravado |

Figura 1 – Taxonomia do conhecimento

Para ajudar a compreender o conhecimento de uma maneira sistemática, [Kim and Park, 2003] apresentaram uma taxonomia do conhecimento (Figura 1). Com efeito, o conhecimento pode ser computorizado ou não. O conhecimento computorizado é o que é armazenado num sistema informático, sendo possível o seu processamento. O conhecimento que não é computorizado está nos cérebros humanos ou é registado em gravadores, mas não é possível o processamento em computador [Kim and Park, 2003].

Registe-se que o conhecimento computorizado explícito é o conhecimento que é recolhido e armazenado em um sistema, tal como bases de dados. O conhecimento computorizado implícito apresenta dois tipos: o conhecimento obtido automaticamente através de ferramentas de *data mining* e o conhecimento dos processos de negócio das aplicações. O conhecimento implícito dos processos de negócio está embebido no fluxo e na lógica do software das aplicações da empresa, como sejam o ERP (*Enterprise Resource Planning*), o SCM (*Supply Chain Management*) e o CRM (*Customer Relationship Management*).

Existem diversos estudos e trabalhos publicados que apontam para a existência de uma relação entre gestão do conhecimento e êxito empresarial. Foi realizado um estudo sobre gestão do conhecimento em 40 empresas do ramo industrial, distribuídas pela Europa, Estados Unidos e Japão. Este estudo foi da responsabilidade da McKinsey & Company e da Universidade Técnica de Darmstadt da Alemanha e visou compreender o modo como a gestão do conhecimento contribui para o sucesso das empresas [Kluge et al., 2002]. Com efeito, foram aplicadas 139 técnicas de gestão do conhecimento, organizadas de acordo com seis características do conhecimento, que se indicam a seguir:

- Subjectividade: a interpretação do conhecimento depende dos conhecimentos anteriores e do contexto em que é utilizado;
- Transferibilidade: o conhecimento pode ser retirado de um determinado contexto e aplicado noutro contexto;
- Sedimentação: o conhecimento não é instantaneamente transferível, ou seja, até certo ponto permanece sedimentado;
- Autovalorização: o conhecimento aumenta em valor quando é partilhado;
- Perecibilidade: o conhecimento está sujeito a uma desactualização à medida que o tempo passa;
- Espontaneidade: o aparecimento de novos conhecimentos não pode ser calendarizado nem previsto com precisão.

Ainda a propósito do estudo Kluge et al. (2002) apresentam-se algumas conclusões:

- Uma gestão do conhecimento bem sucedida é uma poderosa alavanca para o sucesso da empresa. Estabelece-se uma correlação positiva entre um maior sucesso de gestão do conhecimento e a criação de valor na empresa;
- Quanto mais uma empresa usar as boas práticas de gestão do conhecimento, maior será o seu nível de desempenho;
- O conhecimento deve ser distribuído uniformemente por toda a organização, pelo que se sugere a transferência de conhecimento entre departamentos.

[Davenport e Prusak, 1998], indicam um conjunto de princípios sobre a gestão do conhecimento. Estes princípios foram obtidos num estudo efectuado na *British Petroleum* (BP) e são um complemento ao que foi referido. A figura seguinte reúne esses princípios:

| Programa de Trabalho da Equipa BP | Princípios da Gestão do Conhecimento |
|---|---|
| As comunidades do conhecimento foram identificadas e interligadas pela tecnologia | O conhecimento é intrínseco ao ser humano |
| Estabeleceram-se relacionamentos através de reuniões virtuais e presenciais | A partilha de conhecimento exige confiança |
| A tecnologia constituiu-se como um meio para melhorar a comunicação e a colaboração; a formação enfatizou os objectivos e não o hardware e o software | A tecnologia possibilita novos comportamentos ligados ao conhecimento |
| A postura de colaboração das chefias reforçou a importância de novas atitudes | A partilha do conhecimento deve ser estimulada e recompensada |
| As chefias disponibilizaram recursos financeiros e incentivaram a equipa principal | Apoio das chefias é essencial para um programa de gestão do conhecimento |
| Os testes possibilitaram a clarificação dos objectivos definidos | Iniciativas ligadas ao conhecimento devem começar com um projecto-piloto |
| As economias e os aumentos de produtividade foram quantificados favoravelmente | Medições quantitativas e qualitativas são necessárias para avaliar a iniciativa |
| O projecto apesar de ter objectivos específicos deixou espaço à criatividade | O conhecimento é criativo e deve ser estimulado para se desenvolver de forma inesperada |

Figura 2 – Princípios da gestão do conhecimento

Na realidade, a gestão efectiva do conhecimento exige mudanças profundas a nível comportamental, cultural e organizacional. O conhecimento é parte integrante das pessoas, sendo a sua estrutura intelectual complexa e imprevisível. De facto, os dados e a informação são transmitidos electronicamente, mas o conhecimento só consegue transitar de forma eficiente através da rede humana [Davenport e Prusak, 1998]. Assim, regista-se que a tecnologia vem constituir-se como um meio para a identificação, modelação, arquivo e distribuição do conhecimento.

A criação de conhecimento ocorre a partir de uma espiral baseada no compromisso pessoal e nos vários processos de conversão entre o conhecimento implícito e explícito. Envolve o indivíduo, o grupo, a organização e o ambiente [Serrano e Fialho, 2003].

A construção de um repositório de conhecimento numa organização só faz sentido se for disponibilizado aos utilizadores de forma prática e atempada. Isto implica a transformação do sistema de conhecimento num sistema pro-activo, ou seja, sempre que um novo conhecimento é gerado obriga à sua distribuição pelos interessados de acordo com o perfil dos mesmos [Rodrigues, 2000].

O bom senso é um factor importante na problemática da gestão do conhecimento. Deste modo, [Davenport e Prusak, 1998] vem enumerar algumas conclusões que considera serem do senso comum na gestão do conhecimento e que se sintetizam na figura seguinte:

| **Senso Comum na Gestão do Conhecimento** |
|---|
| Importa começar o projecto por uma área de alto valor acrescentado e ter em conta as experiências existentes. |
| Deve-se começar com um projecto-piloto, testar os conceitos e permitir a incorporação de novas iniciativas. |
| É necessário trabalhar em várias frentes simultaneamente, isto é, na tecnológica, organizacional e cultural. |
| Os problemas não devem ser adiados sob pena de se avolumarem irremediavelmente. |
| Os projectos devem estar adequados à cultura organizacional, o que pressupõe uma recolha prévia de elementos antes de se decidir o estilo de actuação e a base de arranque da gestão do conhecimento. |

Figura 3 – Regras de senso comum na gestão do conhecimento

Também Euan Semple [Semple, 2006], com base na sua experiência na BBC, indica algumas regras do que fazer e evitar quando se lidera um programa de gestão de conhecimento (Figura 4).

| **Liderança na Gestão do Conhecimento** |
|---|
| Não falar em gestão de conhecimento. |
| Não gastar demasiado dinheiro. A maior parte desta tecnologia é simples e barata. |
| Criar um sentido de responsabilidade partilhada. Encorajar os utilizadores a serem responsáveis pelo ambiente, como é usado e gerido. |
| Quando as coisas correm mal, fazer perguntas interessantes sobre o "porquê" em vez de ralhar com as pessoas, deixando-as decidir a resposta adequada. |

Seja paciente.

Figura 4 – Regras na liderança da gestão do conhecimento

[Retallick e Sánchez, 1998] vêm reforçar a ideia de que para integrar as comunicações empresariais, tais como as aplicações de processos de negócio e as actividades de desenvolvimento, é necessário informação estrutural. Esta informação é fornecida pelas frameworks San Francisco™ como base para as aplicações de gestão de conhecimento. Por outro lado, estas aplicações ficam disponíveis para outros elementos da equipa de desenvolvimento poderem utilizá-las.

[Llorens e Prieto-Díaz, 2003] clarificam a diferença entre engenharia de conhecimento e gestão de conhecimento, ou seja, para estes autores gestão de conhecimento refere-se à gestão de recursos e ao processo sobre o qual actuam os recursos. A engenharia de conhecimento vem dominar o conhecimento para permitir que os computadores ajudem as pessoas a resolver os seus problemas. Assim, a gestão de conhecimento envolve várias áreas onde se aplicam uma variedade de técnicas. De facto, apesar da engenharia de conhecimento e da gestão de conhecimento gerirem conhecimento, centram-se em aspectos diferentes e complementares.

A gestão de conhecimento recorre a ferramentas de controlo de alterações da engenharia de software na obtenção de tecnologia de apoio com vista à actualização do conhecimento. Na realidade, sem ferramentas de controlo de alterações não é possível conceber uma abordagem para gestão de conhecimento numa organização.

Deste modo, a engenharia de software vem fornecer técnicas de reutilização de conhecimento que são essenciais para organizar o conhecimento. Regista-se por isso uma intervenção ao nível dos modelos de maturidade, modelos de software e garantia de qualidade. Também os factores humanos estão envolvidos neste processo nomeadamente ao nível de novas funções na organização e nos incentivos para efectuar a reutilização. Por fim, importa referir que adaptar conhecimento é o mesmo que reutilizar conhecimento. Assim, as técnicas para a reutilização de conhecimento são efectivamente relevantes para os cientistas do conhecimento.

A gestão de conhecimento assume um papel importante no desenvolvimento de aplicações e particularmente na reutilização. Admite-se que a essência da resolução de problemas assenta na inovação, criatividade, desenho intuitivo, análise correcta e gestão de projectos efectiva envolvendo para isso mais conhecimento tácito que conhecimento explícito. Um dos aspectos do problema e que se refere ao conhecimento explícito, tem a ver com um equívoco que geralmente ocorre entre documentação e entendimento [Highsmith, 2003]. O entendimento refere-se à forma de obter sucesso na reutilização de um componente. De facto, entendimento neste contexto, corresponde a uma combinação de documentação e conversação, sendo a conversação acerca do componente e do contexto no qual opera. Nenhum escritor de documentação pode antecipar todas as perguntas que um utilizador de componente pode fazer. Por outro lado, os sistemas de gestão de conhecimento que se concentraram em reunir, registar e aceder a grandes quantidades de conhecimento, obtidos com recurso a sistemas de interacções de pessoa para pessoa tornam-se muito caros e pouco satisfatórios. Contudo, para um componente complexo, o mais difícil é conseguir o entendimento desse componente a partir apenas da documentação [Highsmith, 2003].

O conhecimento é reutilizado ao longo do processo de desenvolvimento de software, porém conta com um facilitador de entrada e de saída de conhecimento e que é habitualmente designado por porteiro [Fay, 2001]. De facto, esta ideia também é corroborada por [Highsmith, 2003] ao considerar que o conhecimento mais importante

é o que está encapsulado, mas é difícil operar a sua transferência. Sugere a utilização de ferramentas de gestão de conhecimento.

## 3. PAPEL DO GESTOR DO REPOSITÓRIO

No processo de reutilização surge a necessidade de um responsável que incentive a utilização dos artefactos reutilizáveis. Este especialista designado habitualmente por porteiro (*gatekeeper*) actua como gestor ou coordenador do repositório. Assim, as funções principais do gestor do repositório são ilustradas na figura seguinte.

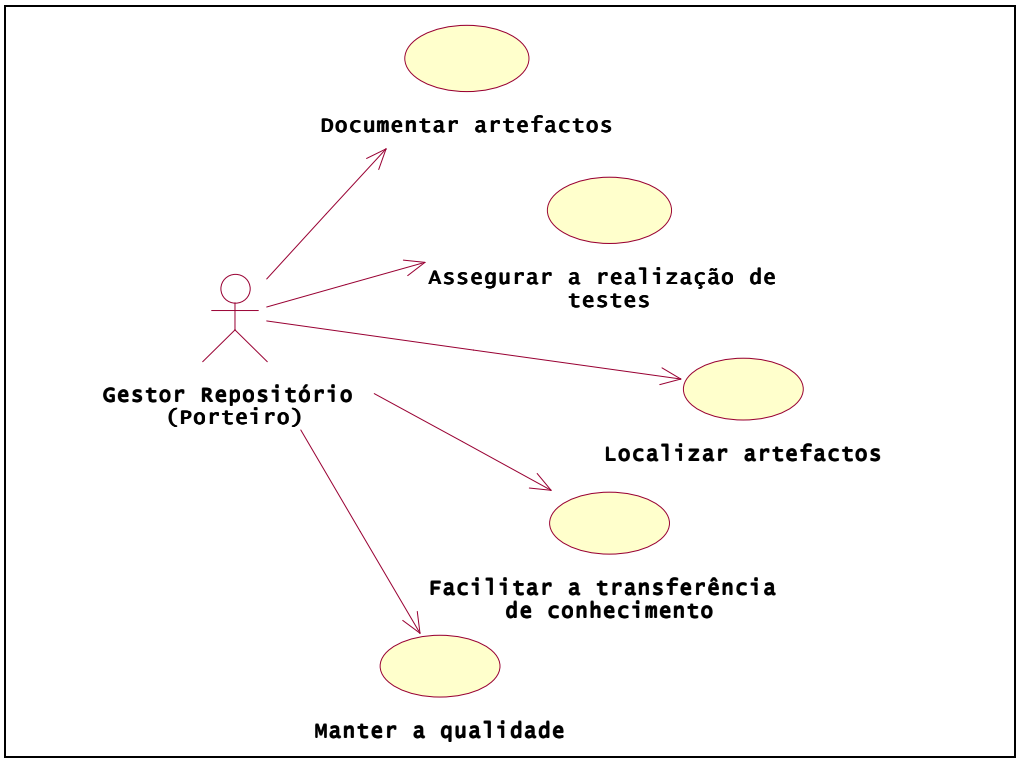


Figura 5 – Funções do gestor do repositório

Este gestor será responsável por coordenar todo o processo nomeadamente na execução das seguintes tarefas:

- Manter a qualidade dos artefactos armazenados dentro do repositório;
- Documentar todos os artefactos com a ajuda dos elementos de desenvolvimento;
- Assegurar que os artefactos de software são testados e que os resultados dos testes são incluídos na documentação;
- Ajudar a equipa de desenvolvimento a localizar os artefactos reutilizáveis;
- Estabelecer contactos entre alguns grupos dispersos;
- Facilitar a transferência de conhecimento.

O tempo e o esforço que a equipa de desenvolvimento investe na pesquisa e recolha de um artefacto reutilizável será em vão se o artefacto estiver incompleto ou inadequado para uma determinada aplicação. Tal facto vem dificultar os esforços para promover a reutilização na organização. Deste modo, o gestor do repositório deve ter

experiência de desenvolvimento, ser um bom comunicador e a ter a capacidade para documentar, organizar e obter informação.

Na realidade, o papel do gestor do repositório é o de facilitar a identificação, a recolha e a transferência de conhecimento. Poderá ainda ajudar a equipa de desenvolvimento a reduzir o tempo gasto à procura de informação vital. Assim, promove-se uma melhoria do trabalho repetitivo e uma diminuição do esforço dispendido.

Uma das formas habitualmente utilizadas para a partilha de conhecimento explícito e tácito é a conversação informal que ocorre entre os elementos da equipa de desenvolvimento, ou seja, trocam-se pontos de vista e outros tipos de conhecimento tácito difíceis de documentar e que são mais facilmente partilhados através de contactos directos. Este facto, vem reforçar o papel do gestor do repositório, uma vez que se torna necessário assegurar que este conhecimento seja armazenado e disponibilizado.

Outra das formas utilizadas pelo gestor do repositório para recolher conhecimento pode ser através de entrevistas e/ou a observação das pessoas no seu ambiente de trabalho. Os resultados obtidos serão posteriormente documentados e disponibilizados.

Aliás, [Davenport e Prusak, 1998] vêm reforçar a necessidade de um porteiro, também designado como corrector ou bibliotecário humano na Internet e que tem a finalidade de localizar material de qualidade. Esta intervenção viria aumentar mais significativamente o valor da Internet como ferramenta do conhecimento do que melhorias puramente técnicas.

A acção do gestor do repositório pode encontrar diversas dificuldades uma vez que há muitos factores culturais que inibem a transferência do conhecimento. Os principais factores correspondem a: falta de confiança entre os intervenientes; diferentes culturas, diferentes vocabulários e quadros de referência; falta de tempo e de disponibilidade para a transferência do conhecimento; falta de reconhecimento do verdadeiro detentor do conhecimento.

As soluções para estas dificuldades são apresentadas por [Davenport e Prusak, 1998]:

- Construir uma rede de contactos que promova a confiança entre os intervenientes;
- Ultrapassar diferenças criando consensos recorrendo à formação e ao trabalho conjunto;
- Criar disponibilidade e formas adequadas para a transferência de conhecimento, como por exemplo promover colóquios, conferências e reuniões;
- Avaliar e recompensar quem partilha e reutiliza o conhecimento.

Finalmente, a implementação de um sistema de incentivo e de reconhecimento a quem reutiliza os artefactos do repositório vem facilitar e aumentar os níveis de reutilização.

## 4. EXEMPLOS DE PADRÕES

As formas utilizadas nos padrões de software variam de autor para autor, mas normalmente possuem as seguintes partes: nome do padrão, descrição breve, contexto, problema, solução, consequências positivas e negativas, exemplo de instanciação do padrão, padrões relacionados, utilizações conhecidas.

Apresentam-se seguidamente alguns exemplos de padrões em formato resumido:

- Padrões identificados por [Biddle et al., 2000] que pretendem melhorar a escrita de casos de uso[1] essenciais (casos de uso críticos ou mais relevantes) e facilitar a especificação de sistemas (Figura 6);
- Padrões desenvolvidos por [Adolph and Bramble, 2003] que fornecem a linguagem necessária para descrever as características de bons casos de uso, ao nível da equipa (Figura 7);
- Padrões para o processo de desenvolvimento [Adolph and Bramble, 2003]. O processo baseia-se no modelo em espiral para incorporar novos dados e descobrir factores de risco de forma iterativa (Figura 8).

| Padrão | Problema | Solução |
|---|---|---|
| Actores | Como é que se deve começar a modelação de casos de uso? | Começando pelas pessoas (ou outros sistemas) que actualmente usam o sistema. |
| Lista de Candidatos a Casos de Uso | Como é que se determina o que o sistema deve fazer? | Listando Candidatos a Casos de Uso para cada Actor. |
| Casos de Uso Essenciais | Como se pode gerir uma grande quantidade de casos de uso? | Escolhendo casos uso essenciais para conduzir o desenho. |
| Diagrama de Casos de Uso | Como é que se sabe que os casos uso estão completos? | Desenhando um diagrama de casos de uso para mostrar quais os actores e os casos de uso que estão relacionados. |
| *Roleplay* de Casos de Uso | Como é que se verifica se os diálogos dos casos de uso estão correctos? | Pondo em prática cada caso de uso antes de ser presente à equipa de desenvolvimento. |

Figura 6– Padrões de casos de uso [Biddle et al., 2000]

Os padrões da Figura 7 indicam-nos que é mais fácil para uma equipa pequena conhecer-se, reunir-se e comunicar para chegar a um consenso. Deste modo, formando equipas pequenas e equilibradas e controlando a participação da audiência obtém-se uma forma de melhorar a qualidade do produto final (casos de uso). Não existe nenhuma forma perfeita que funcione sempre, pelo que, cabe a cada organização de desenvolvimento encontrar o equilíbrio para a equipa funcionar no seu melhor.

| Padrão | Problema | Solução |
|---|---|---|

[1] Caso de uso (*use case*): usado para definir o comportamento externo de um sistema ou entidade semântica sem revelar a sua estrutura interna.

| | | |
|---|---|---|
| *SmallWritingTeam* | Usar muitas pessoas para escrever um caso de uso é ineficiente, e os compromissos feitos para ajustar os diferentes pontos de vista, podem resultar num sistema menos satisfatório. | Restringir o número de pessoas que trabalham num produto para duas ou três. Usar o processo do padrão TwoTierReview (dois tipos de revisões – interna e externa) para incluir mais pessoas. |
| *ParticipatingAudience* | Não se pode satisfazer as necessidades dos *stakeholders* sem a sua participação e contribuição. | Sempre que possível envolver activamente clientes e *stakeholders* internos no processo de desenvolvimento de casos de uso. |
| *BalancedTeam* | Equipas com indivíduos que têm a mesma maneira de pensar na forma de desenvolver casos de uso, pode resultar num conjunto de casos de uso estritamente limitados que não satisfazem as necessidades. | Formar a equipa com pessoas de diferentes especialidades de forma a incentivar os interesses dos *stakeholders* no processo de desenvolvimento. A equipa deve ter programadores e utilizadores finais. |

Figura 7– Resumo dos padrões para a constituição da equipa [Adolph and Bramble, 2003]

O processo de desenvolvimento de casos de uso é descrito por seis padrões resumidos na Figura 8.

| Padrão | Problema | Solução |
|---|---|---|
| *BreadthBeforeDepth* | Não haverá progressos no tempo, nem serão criados de casos de uso coerentes se for gasta a energia a escrever casos de uso muito detalhados. | Direccionar as energias do desenvolvimento de casos de uso para uma visão global e adicionar progressivamente mais detalhes. |
| *SpiralDevelopment* | Desenvolver casos de uso num único passo é difícil, impede a incorporação de novos dados, podendo atrasar a descoberta de factores de risco. | Desenvolver os casos de uso num processo iterativo e em cada iteração aumentar progressivamente a precisão e a objectividade dos casos de uso. |
| *MultipleForms* | Projectos diferentes necessitam de diferentes formalidades durante a sua execução. Pessoas diferentes têm preferências diferentes para o desenvolvimento. Assim, é contra-producente tentar que todas as pessoas usem o mesmo método de trabalho. | Seleccionar um formato de desenvolvimento baseado nos riscos do projecto e nas preferências das pessoas envolvidas. |

| | | |
|---|---|---|
| *TwoTierReview* | Muitas pessoas podem necessitar de rever os casos de uso, tornando este processo caro e demorado. | Assegurar dois tipos de revisões: a primeira efectuada pela equipa a nível interno e repetida várias vezes. A segunda deverá ser realizada pelo grupo completo, podendo fazer só uma única revisão. |
| *QuittingTime* | Desenvolver o modelo de casos de uso para além das necessidades dos *stakeholders*, consome recursos e atrasa o projecto. | Desenvolver os casos de uso até ficarem completos e satisfazerem adequadamente as necessidades da audiência. |
| *WritersLicence* | Uma ênfase excessiva no estilo dificulta a escrita dos casos de uso. | Pequenas diferenças no estilo de escrita são inevitáveis, mas cada caso de uso deve passar nos seguintes testes: deve seguir o modelo e o estilo básico da organização; estar logicamente correcto; ser legível para os avaliadores finais; ser suficientemente preciso para ser implementado. |

Figura 8 – Resumo dos padrões para processo de desenvolvimento [Adolph and Bramble, 2003]

Os padrões fornecem um vocabulário e um entendimento comum para princípios de projecto de software, mas exigem o estudo prévio antes da sua utilização.

## 5. CONSIDERAÇÕES FINAIS

Os padrões representam conhecimento e experiência que pode ser transmitida e ensinada. Podem ainda ser entendidos como boas práticas ou lições apreendidas, uma partilha de experiência (os autores documentam, sob a forma de padrões as boas soluções que observaram terem sucedido em vários projectos). É importante conhecer a existência dos padrões e quanto nos podem ajudar a resolver os problemas.

Neste trabalho defende-se a reutilização de conhecimento através de padrões, nomeadamente padrões para criação / constituição de equipas, documentação de artefactos e definição do processo de desenvolvimento.

Um aspecto importante é o papel do gestor do repositório como facilitador de entrada e saída dos artefactos armazenados. Na realidade, o papel do gestor do repositório é o de facilitar a identificação, a recolha e a transferência de conhecimento. Poderá ainda ajudar a equipa de desenvolvimento a reduzir o tempo gasto à procura de informação vital.

Decorrente da experiência prática é possível concluir que as equipas de desenvolvimento de software reutilizam conhecimento na forma de padrões, componentes, *templates* e outros artefactos de software produzidos por outras equipas. Reutilizar é uma preocupação que deve estar sempre presente no dia-a-dia da equipa.

# BIBLIOGRAFIA

Alexander, C.; Ishikawa, S.; Silverstein, M.; Jacobson, M.; Fiksdahl-King, I.; Angel, S. (1977): *A Pattern Language*; New York, Oxford University Press.

Alexander, C. (1979): *The Timeless Way of Building*. New York, Oxford University Press.

Baskerville, R.; Pawlowski, S.; McLean, E. (2001): Enterprise Resource Planning and Organizational Knowledge: Patterns of Convergence and Divergence.

Biddle, R.; Noble, J; Tempero, E. (2000): Patterns for Essential Use Cases, School of Mathematical and Computing Sciences, Victoria University of Wellington, New Zealand, *Technical Report CS-TR-01/02*, http://www.mcs.vuw.ac.nz/research.

Booch, G. (1998): Patterns; *The Patterns Handbook: Techniques, Strategies, and Applications*; Rising, Linda (Ed); Cambridge University Press, pp. 237 – 241.

Buschmann, F.; Meunier, R.; Rohnert, H.; Sommerlad, P.; Stal, M. (1996): *Pattern Oriented Software Architecture - a System of Patterns*; John Wiley and Sons.

Coad, P. (1992): Object-Oriented Patterns; *Communications ACM*; 35, 9, pp. 152-159.

Coplien, J.; Schmidt, D. (Eds.) (1995): *Pattern Languages of Program Design*; Addison-Wesley Publishing Company.

Fay, Sharon (2001): The Gatekeepers; Flashline, September, disponível em http://www.flashline.com/content/Fay/fay2.jsp.

Figueiredo, Dias (2003): Gestão do Conhecimento e Economias de Inovação Intensiva; *Gestão de Empresas na Era do Conhecimento*, Silva, R.; Neves, A.(Org.); pp. 435-441, Edições Sílabo.

Fowler, M. (1997): *Analysis Patterns: Reusable Object Model*; Addison-Wesley.

Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. (1995): *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley.

Highsmith, Jim (2003): Reuse as a Knowledge Management Problem; Disponível em: http://www.awprofessional.com/articles/article.asp?p=31478.

Jones, M. C.; Cline, M.; Ryan, S.; (2006) : Exploring Knowledge Sharing in ERP Implementation: An organizational Culture Framework; Decision Support Systems; Vol. 41, pp. 411–434.

Llorens, J.; Prieto-Díaz, R. (2003): Is KS=(D+I+S+K)*E + KM?; ACM SIGSoft Software Engineering Notes, Vol. 28, Nº 2; Disponível em: https://users.cs.jmu.edu/prietorx/Public/publications/KS_DISK_EKM-final.doc.

Keepence, B.; Mannion, M. (1999). Using Patterns to Model Variability in Product Families; IEEE Software; Vol. 16, Nº 4, pp. 102-108.

Kim, Won; Park, Seung-Soo (2003): *Knowledge Management: A Careful Look*; Journal of Object Technology, Vol. 2, Nº 1, pp. 29-38; http://www.jot.fm/issues/issue_2003_01/column4.

Kluge, Jürgen; Stein, Wolfram; Licht, Thomas (2002): *Gestão do Conhecimento: Segundo um Estudo de McKinsey & Company*; Trad. Sofia Barreto Leitão; Cascais, Principia-Publicações Universitárias e Científicas.

Larman, Craig (2002): *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process*, Prentice-Hall Inc., Second Edition.

Monday, Paul; Carey, James; Dangler, Mary (1999): *SanFrancisco Component Framework: An Introduction*; Addison Wesley Longman.

Meszaros, G. (1996): A Pattern Language for Improving the Capacity Of Reactive Systems; Pattern Languages of Program Design 2; Vlissides, J., Coplien, J., Kerth, N. (Eds.); Cap. 35, pp. 575-591, Addison-Wesley Publishing Company.

Retallick, R.; Sánchez, S. (1998): Enterprise Knowledge Sharing, Activity Management, and a Fabric for Commitment; San Francisco Frameworks; IBM Systems Journal, Vol. 37, Nº 2. Disponível em http://www.research.ibm.com/journal/sj/372/retallick.html.

Rising, Linda (Ed) (1998). The Patterns Handbook: Techniques, Strategies, and Applications; Cambridge University Press.

Semple, E. (2006): *Entrevista de Euan Semple* a Ana Neves, http://www.kmol.online.pt/pessoas/SempleE/entrev_1.html, Consultado em 29.11.2006.

Serrano, A.; Fialho, C. (2003): Gestão do conhecimento: o novo paradigma das organizações; Lisboa, FCA Editora de Informática.

Silva, R.; Neves, A. (Org.) (2003): Gestão de Empresas na Era do Conhecimento; Edições Sílabo.

Silveira, C. (2003): O Impacto da Reutilização em Projectos de Melhoria do Processo; *Memórias 2da. Conferencia Iberoamericana en Sistemas, Cibernética e Informática CISCI 2003*, Orlando, EUA, Vol.I, pp. 224-229.

Simon, Herbert A. (1975): The Sciences of the Artificial; Cambridge, MA, The M.I.T. Press.

Vlissides, J., Coplien, J., Kerth, N. (Eds.) (1996): *Pattern Languages of Program Design 2*; Addi-son-Wesley Publishing Company.