Engenharia de Requerimentos

Culpada por 80% das falhas em projetos,
a disciplina merece melhor trato.

Requirements Engineering

Blamed by 80% of the project fails,
the discipline deserves better care.

Paulo Fernando Vasconcellos Nogueira
Rua Afonso Celso, 1.043 Apto. 111
CEP 04.119-061 – Vila Mariana
São Paulo – SP
pfvasconcellos@yahoo.com

Nunca antes, em sua breve história, as organizações de TI tiveram tamanha e tão crítica importância. Quando sua principal função limitava-se a sistemas de controle, sua fama de *caixa-preta* se justificava. Mas o mundo mudou e empresas de todo porte e natureza começaram a perceber a importância estratégica dos sistemas de informação e dos projetos que os definem e implementam. Descobrimos, enfim, que os projetos de TI têm dois grandes significados para as modernas organizações: i) Eles representam, obrigatoriamente, MUDANÇAS; e ii) Eles são a melhor oportunidade de consolidação de CAPITAL INTELECTUAL. Porém, infelizmente, novos e bonitos rótulos (e responsabilidades) não representaram avanços na qualidade dos projetos. Pelo contrário, os pecados, atrasos e *bugs* podem ser testemunhados ao redor do globo. Eles ficaram mais visíveis e comprometedores após o advento da *Internet* comercial. Dados recentes mostram que projetos de TI geralmente custam duas vezes seu valor original. Um terço deles é cancelado. Carecemos de estatísticas *tropicalizadas*, mas a realidade brasileira deve ser a mesma, se não for pior. Quais nossas deficiências? O que torna os projetos de TI tão frágeis e problemáticos? Há cura?

Este artigo NÃO tem a ambição de apresentar todas as respostas para as questões acima. E, definitivamente, NÃO será um *check-list* miraculoso. No entanto, trataremos aqui de uma disciplina que, segundo algumas pesquisas, responderia por 80% das causas de fracassos em projetos de TI: a Engenharia de Requerimentos (RE – Requirements Engineering).

Segundo entusiastas, a RE é a mais importante e menos compreendida das sub-disciplinas da Engenharia de Software. E a confusão nasce na própria explicação do termo "requerimento" (ou requisito). Na definição clássica um requerimento é "uma condição ou funcionalidade requerida por um usuário para resolver um problema ou atender determinado objetivo". Qual o problema?

A Voz do Usuário

O usuário fala ou apresenta mais que "condições" e "funcionalidades desejadas". A definição abstrata e ambígua alimentou procedimentos muito desestruturados. Gerou ferramentas que utilizam editores de texto como base para gestão de requerimentos. Tamanha abstração e falta de estrutura explica a distância entre código e requerimentos, o que normalmente resulta em:

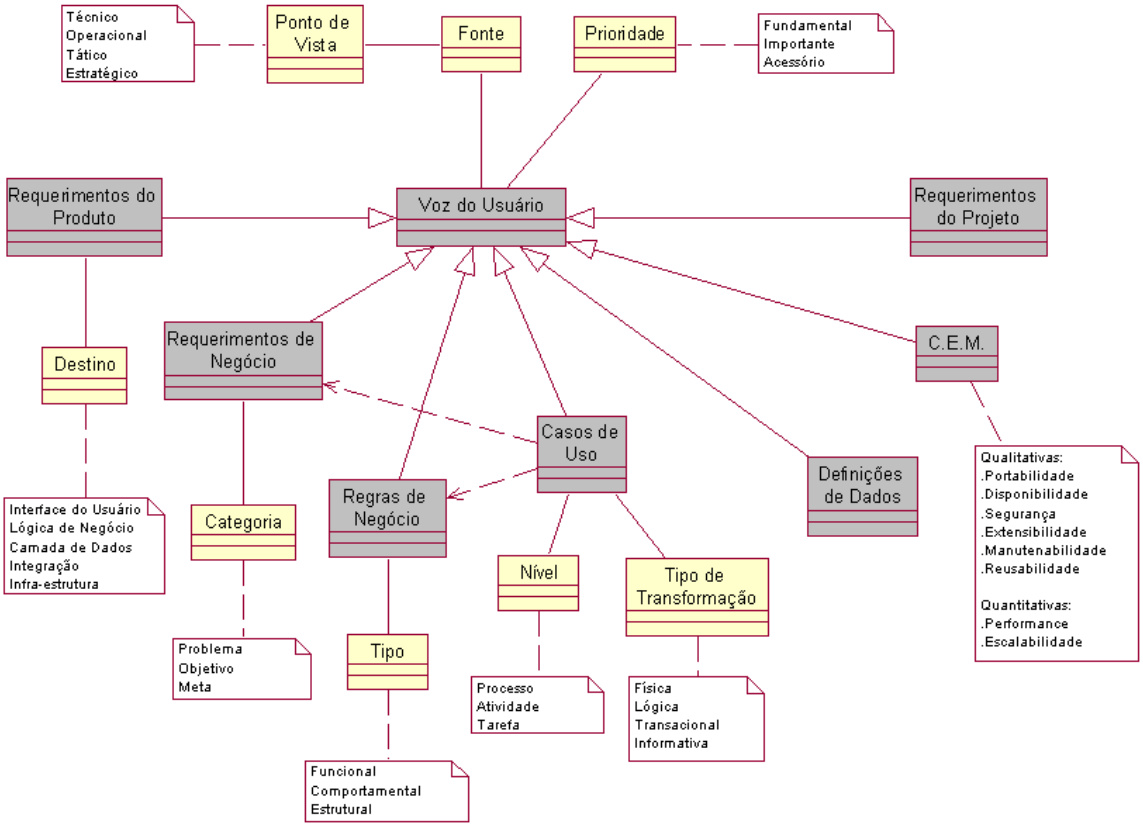
- Má compreensão daquilo que o usuário realmente precisa;
- Fraca definição dos requerimentos, muitas vezes confundida com excesso de volatilidade; e
- Péssima gestão de mudanças.

Alcançamos nosso primeiro ponto-chave: Devemos tratar os requerimentos de uma forma mais rica.

Por mais confiável que seja a memória de nossos analistas, devemos buscar a transformação de todas as informações obtidas junto aos usuários em "conhecimento explícito" (veja *Box* "Capital Intelectual"). Chega a ser contraditório o fato de não buscarmos a máxima estruturação dos requerimentos. Por estruturação entenda-se sua quebra em atributos que facilitem sua compreensão, avaliação, priorização, recuperação e, principalmente, sua realização. A voz do usuário deve se encaixar em uma das classes abaixo:

- **Requerimentos do Produto**: descrevem uma característica ou funcionalidade que o produto a ser gerado deve oferecer;
- **Requerimentos de Negócio**: metas ou objetivos aplicados a indicadores de negócio que devem ser afetados (positivamente, de preferência) pela solução;
- **Regras de Negócio**: 'ordens' que definem ou restringem algum aspecto do negócio;
- **Casos de Uso ou Cenários**: a descrição de um Processo de Negócio, Atividade ou Tarefa;
- **Definições de Dados**: que vão desde a macro-visão da arquitetura de informações até detalhes como o tipo de um determinado campo;
- **Características Essenciais Mínimas (C.E.M.)**: atributos de qualidade que devem ser atingidos pelo sistema em construção;
- **Requerimento do Projeto**: tarefas, restrições ou regras que afetam a gestão do projeto.

A figura abaixo mostra uma visão conceitual de uma base de conhecimentos estruturada:



A 'voz do usuário', ao ser registrada, deve obedecer (cegamente) algumas regras:

- Expressar uma única informação (necessidade), preferencialmente em uma sentença apenas;
- Não permitir diferenças de interpretação (evitar ambigüidade);
- Controlar versões (mudanças). Uma vez registrado o requerimento não deve ser apagado.

Devemos obrigatoriamente registrar a 'Fonte' do requerimento. Quando esta for uma pessoa devemos também registrar seu ponto de vista, que pode ser:

- **Técnico:** equipe de desenvolvimento ou de TI;
- **Operacional:** executor do processo de negócio (ou de parte dele);
- **Tático:** gerente, supervisor ou qualquer outra função de nível médio, responsável direto pela administração do processo de negócio ou atividade em questão;
- **Estratégico:** topo da pirâmide hierárquica, normalmente representado por presidentes, diretores, VP's ou afins.

Outro atributo de suma importância é a 'Prioridade' do requerimento. Aqui, como na redação do requerimento, devemos ser práticos e objetivos. Um requerimento só pode estar em uma das três categorias abaixo:

- **Fundamental:** deve ser atendido, ponto. Sua não satisfação significa falta grave do projeto;
- **Importante:** o usuário saberá viver (por um tempo) sem sua implementação;
- **Acessório:** que costumamos chamar de 'perfumaria'.

A existência dessa classificação facilita a gestão de escopo e prazos do projeto. A classificação de um requerimento deve ser fruto do consenso entre usuários e desenvolvedores.

Requerimentos do Produto e C.E.M.'s

Quando registramos um 'Requerimento do Produto', devemos apontar sua possível destinação. No diagrama acima direcionamos o requerimento para uma das 5 células de especialização que existiriam na organização (ou equipe do projeto). São elas:

- **Interface do Usuário:** ou Camada de Apresentação;
- **Lógica de Negócio:** a Camada Intermediária em uma arquitetura cliente/servidor *n* camadas;
- **Camada de Dados:** que preferimos chamar de Arquitetura de Informações, por ser responsável por dados estruturados (SGBD's) ou não (CMS's – Gerenciadores de Conteúdo);
- **Integração:** parte da equipe que seria responsável pelas 'pontes' entre sistemas; e
- **Infra-estrutura:** os responsáveis pelo planejamento e implementação de todos os elementos de infra-estrutura (*hardware* e *software*) requeridos pela solução.

No momento do registro dos requerimentos dessa natureza devem aparecer (explicitamente ou não) atributos de qualidade e algumas métricas fixadas pelos usuários. Esse tipo de requerimento é o que chamamos de 'Características Essenciais Mínimas' (C.E.M.). Quando específicas de uma parte da solução, as necessidades devem ser registradas como 'Requerimentos do Produto'. No entanto, seu conjunto deve ser registrado como uma CEM. Por exemplo: *"o sistema deve ser capaz de atender, simultaneamente, 500 usuários"*. Essa é uma CEM, especificamente o atributo de qualidade "Escalabilidade".

Desvendando o 'Negócio'

O 'negócio' é registrado em três categorias distintas de requerimentos: Casos de Uso, Requerimentos de Negócio e Regras de Negócio. No primeiro, registramos cenários de utilização, descrevendo tanto a situação atual (*as is)* como o futuro desejado *(to be)*. Seguindo uma visão tradicional obtemos três níveis de casos de uso: os Processos de Negócio, Atividades e Tarefas (além das especificações de casos de uso podemos utilizar Diagramas de Atividades da *UML,* que favorecem tremendamente a legibilidade dos fluxos). Os processos, atividades ou tarefas possuem atributos específicos. São eles: Objetivo, Propósito, Documentação, Dono do Processo, Atores, Prioridade, Riscos, Possibilidades de Melhoria, Tempo e Custo. A obtenção ou não de tais informações depende muito da característica do projeto. Projetos que obriguem o redesenho de processos de negócio devem se preocupar com o registro de tais informações, facilitando análises do retorno sobre o investimento *(ROI)* e estudos de viabilidade.

O excelente livro *"Business Modeling with UML – Business Patterns at Work"*, de *Hans-Erik Eriksson* e *Magnus Penker (Wiley – 2000)*, propõe uma maneira ainda mais detalhada e ampla para a documentação de processos de negócio (lançando mão de vários diagramas e elementos *UML* para tal). Trata-se de um complemento (tanto do processo quanto da notação) altamente recomendado para projetos de médio e grande porte e para a documentação da 'Arquitetura de Negócios'.

'Requerimentos de Negócio', que podem ser objetivos ou metas, devem ser vinculados diretamente aos casos de uso. Objetivos e metas também são alocados de forma hierárquica, a exemplo do que podemos fazer com os 'requerimentos do produto'. Em alguns projetos tais requerimentos serão coletados para a geração de um contrato de risco ou para a definição de níveis de serviço.

As 'Regras de Negócio' são registradas individualmente, apesar de serem relacionadas com os casos de uso capturados. Possuem atributos específicos, como o tipo (Funcional, Comportamental ou Estrutural) e categoria (Derivação, Restrição ou Existência).

Process, what Process?

RUP ou *eXtreme Programming?* Não importa seu gosto, necessidade (ou religião). Seja através de casos de uso *(RUP)* ou *User Stories (XP)* estamos de alguma forma executando a engenharia de requerimentos. Independente da formalidade do processo, as atividades abaixo serão executadas:

- **Coleta:** a captura e registro dos requerimentos (da 'voz do usuário');
- **Análise:** o estudo dos requerimentos coletados;

- **Derivação:** a transformação dos requerimentos em outro tipo de conhecimento explícito (modelos ou código).

No primeiro passo apenas nos preocupamos em registrar todas as solicitações (além de entendê-las, é claro). O cadastramento dos requerimentos deve ocorrer após as reuniões de coleta, já que, dependendo da estrutura da base (e da ferramenta utilizada), pode ser um trabalho demorado.

A análise dos requerimentos pode ocorrer na sequência. Em um ciclo incremental e iterativo é altamente recomendado que a análise sempre ocorra imediatamente após a conclusão da coleta. O principal produto gerado nesta etapa é uma grande matriz de rastreabilidade na qual confrontamos todos os requerimentos entre si. Com a estruturação proposta acima conseguimos facilitar a análise, já que podem ser geradas várias matrizes (Casos de Uso X Requerimentos de Negócio, Requerimentos de Negócio X Requerimentos do Produto, Casos de Uso X Regras de Negócio, CEM X Requerimentos do Produto, etc). Existem 5 tipos de relacionamentos possíveis entre os requerimentos:

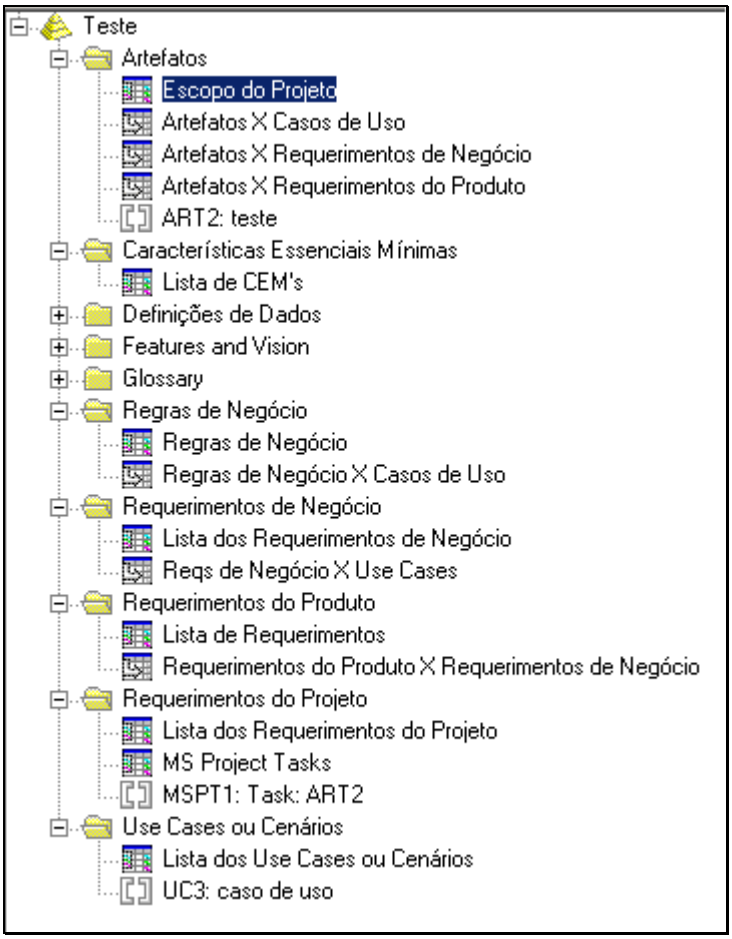
- **Indiferente:** o nome diz tudo;
- **Dependente:** a satisfação do requerimento *B* depende da satisfação do requerimento *A* (é assim que registramos a hierarquia de todos os requerimentos, independente de sua classe);
- **Complementar:** a satisfação do requerimento *A* facilita a satisfação do requerimento *B* (porém a satisfação do requerimento *A* não é obrigatória);
- **Substituto:** o requerimento *A-2* substitui o requerimento *A-1* (sim, a gestão de mudanças começa aqui!);
- **Conflitante:** a satisfação do requerimento *A* impede a satisfação do requerimento *B*.

A relação de dependências e complementaridades favorece a geração de um plano de projeto (e da *WBS, work-breakdown structure*). O controle de substituições é o núcleo da gestão de mudanças em um projeto. A detecção dos requerimentos conflitantes facilita a antecipação de problemas e sua comunicação. Apesar da descrição aqui ser um tanto simplista, trata-se de um processo burocrático, delicado, mas de suma importância para a execução do projeto. Aqui começa a se justificar o nome "Engenharia de Requerimentos".

Casa de ferreiro...

Tamanha base de conhecimentos não pode ser tratada com 'espetos de pau e papel jornal'. Não é difícil criar uma base que pode ser implementada em um pequeno gerenciador de base de dados, como o *MS Access*. Podemos optar por customizar um produto de mercado criado especificamente para a gestão de requerimentos, como o *Doors* da *Telelogic* ou o *RequisitePro* da *Rational*. A

figura abaixo mostra a *treeview* do *RequisitePro* com a implementação de toda a estrutura proposta.



A consolidação da 'voz do usuário' em uma única base facilita seu gerenciamento. Perfis podem ser criados para que analistas, coordenadores e desenvolvedores atuem exclusivamente com as informações que lhes são específicas. Devemos buscar também a total rastreabilidade dos requerimentos. A indicação de qual artefato, na maior granuralidade possível, satisfaz determinada necessidade é crucial na condução de um projeto. Nos produtos da *Rational,* tal rastreabilidade é possível com a construção de um *add-on* para o produto *Rational Rose*. "Enxergando" a base do *RequisitePro* (que pode ser implementada como um arquivo MDB ou nos produtos *Oracle e MS SQL Server)*, podemos apontar uma classe ou mesmo atributos e métodos para determinados requerimentos. Os elementos do *Rose* seriam registrados como 'Artefatos' na base do *RequisitePro*. O vínculo, portanto, existiria na própria base de dados deste produto. O *Rose* oferece uma interface mais natural para analistas e desenvolvedores. Aproximando-o da base de conhecimentos do projeto favorecemos a produtividade da equipe e a gestão do escopo do projeto.

Choque de Realidade

O *Software Engineering Institute (SEI)* propõe em seu *site* um *reality check* para projetos de desenvolvimento de sistemas:

- ✓ Qual meta ou objetivo de negócio é atendido pelo requerimento?
- ✓ Onde o requerimento é satisfeito (está implementado)?
- ✓ Este requerimento é realmente necessário?
- ✓ Como eu devo interpretar este requerimento?
- ✓ Quais decisões de projeto afetam a satisfação deste requerimento?
- ✓ Por que foi implementado desta forma e quais eram as alternativas?
- ✓ Este elemento é realmente necessário?
- ✓ Qual teste de aceitação será utilizado para validar o requerimento?
- ✓ Qual o impacto gerado pela mudança deste requerimento?
- ✓ Todos os requerimentos foram atendidos?
- ✓ O projeto acabou??

Com certeza a existência de uma base de conhecimentos consolidada (e sua integração com o ambiente de desenvolvimento, ferramentas *CASE* e *IDE's)* facilita a resposta desse questionário tão agressivo (e, por que não dizer, incômodo).

Capital Intelectual

Foi o *best-seller* de Thomas Stewart, "Capital Intelectual", que colocou esse conceito na agenda das grandes corporações. Existem duas grandes categorias de conhecimento: o TÁCITO, que reside na cabeça dos colaboradores, parceiros e clientes; e o EXPLÍCITO, saber que está documentado de alguma forma. O conhecimento explícito seria apenas 1% de todo o capital intelectual das empresas!

Fatos:

1. O *software* e todos os seus artefatos periféricos (diagramas *UML*, código-fonte, etc) são conhecimento explícito. São um tipo de capital intelectual que chamamos de ESTRUTURAL. Deveria ser tratado como patrimônio;
2. Um projeto de desenvolvimento de *software* é uma das melhores oportunidades de EXPLICITAÇÃO, ou seja, a transformação de conhecimento tácito em conhecimento explícito. Estamos aproveitando tamanha oportunidade? Documentação é realmente só uma burocracia a ser ignorada por analistas e desenvolvedores?

REFERÊNCIAS:

Rational Unified Process – www.rational.com/rup
Software Engineering Institute (SEI) – www.sei.cmu.edu
Capital Intelectual / Thomas Stewart – members.aol.com/thosstew

BIBLIOGRAFIA:

Software Requirements
Wiegers, Karl E. (Microsoft Press) - 1998
Requirements Engineering – A Good Practice Guide
Sommerville, I. e Sawyer, P. (John Wiley & Sons Ltd.) – 1997
Business Modeling with UML – Business Patterns at Work
Penker, Magnus e Eriksson, Hans-Erik (Wiley) – 2000
Winnig with Software – An Executive Strategy
Humphrey, Watts S. (Addison-Wesley) - 2001
Capital Intelectual – A Nova Vantagem Competitiva das Empresas
Stewart, Thomas (Editora Campus) – 1998
Criação de Conhecimento na Empresa
Nonaka, Ikujiro e Takeuchi, Hirotaka (Editora Campus) – 1997

Paulo F. Vasconcellos Nogueira
pfvasconcellos@yahoo.com