Impactos Organizacionais da Gestão de Projetos em Pequenas Empresas de Software

Autoria: Cicero Aparecido Bezerra

Resumo

O desenvolvimento de sistemas de informação tem apresentado vários fatores que contribuem para a insatisfação do cliente. Quando se verificam os problemas citados pelos usuários, encontram-se muito comumente, fatores como o não cumprimento de prazos estabelecidos, orçamentos imprevisíveis e não funcionalidade. Por outro lado, a equipe de engenheiros de software convive com prazos mal dimensionados, orçamentos estimados abaixo da real necessidade e com especificações de sistemas incompletas. A análise dos problemas levantados pelos atores presentes nesta situação, sob uma perspectiva mais ampla, pode permitir chegar à conclusão de que o problema reside, não especificamente em ferramentas ou métodos de desenvolvimento de software, mas sim no seu processo de gerenciamento. O presente artigo avalia os resultados da implantação de gerenciamento de projetos em uma pequena software-house, sob a ótica dos engenheiros de software, nos aspectos de produtividade, controle, satisfação e inovação. Esta avaliação é efetuada em 2 momentos: imediatamente à conclusão do primeiro software desenvolvido e gerenciado formalmente e após transcorridos 2 anos, com a intenção de verificar se o processo pode ser continuado, com sucesso, ou se trata apenas de mais uma metodologia passageira que não agrega valor ao produto final.

PALAVRAS-CHAVE: gestão de projetos, pequena empresa, desenvolvimento de software.

1. INTRODUÇÃO

Assim como na literatura, a experiência prática tem mostrado que o desenvolvimento de produtos informáticos apresenta problemas facilmente perceptíveis, tanto pelo usuário final, como pelo engenheiro de software. No intuito de solucionar estes problemas, novos modelos, metodologias, técnicas e ferramentas de desenvolvimento são criadas e aplicadas efusivamente. Apesar disto, os problemas vêm persistindo. Pode-se concluir, ainda que superficialmente, que as ocorrências de erro não residem, exclusivamente, nos instrumentos e/ou metodologias empregados no desenvolvimento, devido justamente à gama de opções existentes.

Portanto, é natural buscar novas variáveis para equacionar o problema. Ao analisar o processo de engenharia de software, de maneira sistêmica, percebe-se que as falhas não estão vinculadas diretamente a esta ou aquela fase do ciclo de desenvolvimento. Assim sendo, ao constatar que o gerenciamento de projetos está (ou deveria estar) presente em todas as etapas, torna-se natural voltar a atenção neste aspecto. A hipótese de que os problemas existentes no software estão mais diretamente vinculados à sua gestão, poderia ser comprovada através dos fatores apontados pelo usuário final, como problemas: entrega fora do prazo, custos excessivos e não funcionalidade.

Nesta perspectiva enquadra-se o estudo apresentado: verificar a validade da utilização da gestão formal de projetos, aplicada em uma pequena software-house, abordando um pequeno referencial teórico (abrangendo autores clássicos e contemporâneos) nas questões relacionadas ao processo de desenvolvimento de software nas pequenas empresas e gerenciamento de projetos. O trabalho ainda irá descrever, resumidamente, o processo de implantação e analisar quantitativamente os impactos percebidos pela equipe (nos aspectos produtividade, controle, satisfação e inovação), proporcionados pelo gerenciamento de projeto no ciclo de desenvolvimento de software. Além disso, o estudo irá comparar os resultados transcorridos 2 anos após a implantação da gestão de projetos na software-house estudada, verificando se o tema agrega valor, provocando rupturas tecnológicas, ou se simplesmente tratou-se de um modismo passageiro.

2. REFERENCIAL TEÓRICO

Neste tópico será apresentada a pesquisa bibliográfica no que se refere ao processo de desenvolvimento de software, o gerenciamento deste processo e sua importância, além de um breve perfil das pequenas empresas desenvolvedoras de sistemas informatizados, com o objetivo de fornecer elementos necessários para o entendimento de seus conceitos básicos. É importante salientar que não se pretende esgotar o assunto, mas sim identificar os aspectos principais para a para a contextualização do tema abordado neste estudo.

2.1 Engenharia de software

Há muito tempo questiona-se sobre a falta de qualidade em software. De acordo com BEZERRA (p.40, 2001), além (evidentemente) da funcionalidade do software, o cliente visualiza a qualidade em aspectos de custos e prazo de entrega. Neste sentido, o STANDISH GROUP (p.73, 2003), realizou estudos nos anos de 2000 e 2001 nos Estados Unidos e constatou que 72% dos projetos de software entregues apresentavam problemas, mais especificamente em não cumprimento de prazos e custos e falhas em alcançar a funcionalidade requerida pelo usuário. No Brasil, a observação empírica leva a acreditar que a porcentagem de problemas é, no mínimo, igual àquela encontrada nos Estados Unidos.

O processo de desenvolvimento de software tem ganhado um aspecto mais formal e científico, a partir da constatação dos problemas mencionados anteriormente, principalmente depois que o termo "engenharia" passou não somente a ser utilizado, mas também aplicado na sua essência. De acordo com PETERS e PEDRYCZ (p.5, 2000), uma das justificativas da aplicação da abordagem de engenharia na construção de software é a produção de sistemas informatizados que respeitem prazos e orçamentos (além, obviamente, dos aspectos funcionais). Para PAULA FILHO (p.6, 2001), como todo produto industrial, o software é concebido a partir da percepção da realidade, desenvolvido, colocado em operação e finalmente, descartado. Com poucas variações entre autores (PAULA FILHO, p.18, 2001; PRESMAN, p.33, 2001; PETERS e PEDRYCZ, p.41, 2001; MARTINS, p.108, 2002), o processo de desenvolvimento é composto pelas seguintes etapas:

- Requisitos: definição do domínio da informação e função do software, desempenho e interfaces. O objetivo é identificar as características técnicas, funcionais e estruturais do sistema, conforme observada pelo cliente.
- Análise: elaboração de um modelo conceitual estático (dados) e dinâmico (funcionalidades). Várias metodologias podem ser abordadas nesta fase (análise estruturada, análise essencial, análise orientada a objetos, entre outras) com o objetivo de representar o negócio em si, subtraindo características relacionadas à tecnologia empregada na construção do software.
- Projeto: nesta etapa, agrega-se ao projeto (*design*), as características tecnológicas ao qual será submetido: linguagem de programação, banco de dados, sistema operacional, rede, interfaces com os usuários, entre outros, conforme MARTINS (p.118, 2002).
- Codificação: é a tradução do projeto (*design*) para uma linguagem de programação. Conforme PRESSMAN (p.35, 2001), caso o projeto tenha a profundidade técnica adequada, a etapa da codificação será efetuada mecanicamente.
- Testes: à medida em que o software está sendo codificado, passa por testes que visam garantir a integridade de seus aspectos lógicos internos, bem como sua funcionalidade (aspectos externos).
- Implantação: depois de um programa de treinamento aos usuários finais, finalmente o software é instalado no ambiente de produção, passando por um acompanhamento (para avaliar sua performance).

A estas etapas, pode-se acrescentar a manutenção, responsável pela readequação do sistema ao longo do tempo, conforme existirem alterações de requisitos. É importante frisar que entre

estas etapas, existem tarefas que podem ser projetadas de modo a serem executadas simultaneamente, garantindo agilidade no processo, conforme BEZERRA (p.97, 2001).
Finalmente, há que se destacar o papel da gerência de projetos na engenharia de software. De acordo com PRESSMAN (p. 55, 2001), o processo da gestão está presente em todas as etapas da elaboração do software. Sua importância está intimamente associada à qualificação da empresa, sendo a etapa intermediária entre níveis caóticos de desenvolvimento e níveis cuja característica contemplem com o processo integrado de engenharia de software, preconizado no modelo CMM (*Capability Maturity Model*), conforme FIORINI, STAA e BAPTISTA (p. 20, 1998).

2.2 Gestão de projetos

De acordo com BELLOQUIM (p.13, 2002), nenhuma técnica de desenvolvimento de software terá resultados positivos se não forem adotados processos ordenados (formais) de gerenciamento de projetos. Devido à característica do ciclo de vida do desenvolvimento de um produto de software, sua concepção enquadra-se no conceito de projeto que, segundo CASAROTTO FILHO *et al* (p.19, 1999) é "um conjunto de atividades interdisciplinares, interdependentes, finitas e não repetitivas" visando atingir um objetivo pré-estabelecido. Este conceito é complementado por MARTINS (p.4, 2002) ao introduzir o elemento "incerteza" na realização das atividades, bem como ao especificar o envolvimento de pessoas, prazos, custos e escopo. É importante contextualizar a palavra "projeto" empregada neste estudo. De acordo com CASAROTTO FILHO *et al* (p.19, 1999), o termo pode estar relacionado a um conjunto de planos, especificações e desenhos de engenharia (o correspondente na língua inglesa é *design*). Já, para o conceito apresentado (atividades, objetivos, cronograma, orçamento), seu correspondente na língua inglesa é *project* e será este o enfoque empregado no decorrer do trabalho. Assim sendo, é importante caracterizar o ciclo de vida de um projeto (genérico) que, segundo VALERIANO (p.23, 1998), é composto pela execução de quatro fases:

- Conceptual: idéia inicial do produto, elaboração e aprovação de proposta.
- Planejamento e organização: o projeto é decomposto em etapas, planejado e organizado de forma a evidenciar os aspectos de controle e execução.
- Implementação: as etapas são desenvolvidas, sob uma coordenação formal, até a obtenção dos objetivos propostos.
- Encerramento: efetivação da transferência dos resultados ao cliente, avaliação geral do projeto (ao que pode-se especificar a determinação de fatores críticos de sucesso e pontos que ficaram aquém do esperado) e desmobilização da equipe e recursos alocados ao projeto.

Estas fases não são estáticas, nem seqüenciais. Elas se sobrepõem durante todo o ciclo de vida do projeto, com predominância de uma ou outra de acordo com o andamento dos trabalhos, conforme VALERIANO (p.24, 1998).
No caso de implementação de produtos de software, algumas metodologias de gererenciamento vêm sido empregadas. Dentre elas, pode-se citar o *Unified Software Development Process*, que segundo MARTINS (p.108, 2002), regulamenta o processo de gerenciamento do projeto de desenvolvimento de sistemas e parte do pressuposto de que os desenvolvedores não dispõem de todas as exigências no início do ciclo de vida do software e, de acordo com QUATRANI (p.5, 2000), devendo ser previstas mudanças em todas as fases, tornado o processo interativo e incremental, ao que pode-se incluir, atividades executadas em paralelo. Na realidade, o desenvolvimento simultâneo de algumas etapas que compõem o ciclo de vida do software tem se mostrado bastante produtivo e, conforme relatado por BEZERRA (p.97, 2002), atingiu índices elevados de satisfação do usuário final (qualidade). Porém, a linha que separa a engenharia (simultânea), do "artesanato" (implementação caótica) de software, é o gerenciamento formal do projeto de desenvolvimento. A aplicação do *Unified Software Development Process* terá maiores chances de resultar em sucesso se agregar

algumas outras ferramentas de controle às interações. Neste sentido, a utilização do modelo PERT/CPM pode produzir índices bastante satisfatórios, refletidos tanto no projeto, como no produto final.
Finalmente, sobre a gestão de projetos de software, ainda há muito o que se estudar, uma vez que, em um primeiro momento, é necessário medir os processos e, neste sentido, não existe uniformidade em padrões de medidas capazes de quantificar todos os aspectos que envolvem o desenvolvimento de sistemas informatizados, causando uma verdadeira "guerra de métricas". Esta situação, identificada por PETRINI e POZZEBON (p.2, 2000), decorre do fato de que, a cada nova pesquisa, surge um novo modelo conceitual, minimizando esforços para validar os modelos existentes. Ainda que tal atitude promova riqueza e diversidade, limita o amadurecimento da área como disciplina científica. É importante salientar que, apesar da falta de uniformidade (ou de um padrão), a utilização de quaisquer métricas já é um ponto positivo (e essencial) no gerenciamento de projetos.

2.3 Desenvolvimento de software na pequena empresa

As software-house's de pequeno porte diferem de outras pequenas empresas apenas no seu processo produtivo. Para HARRIS (p.8, 1999), estas organizações são caracterizadas por serem operacionalizadas pelo proprietário, possuírem poucos empregados e poucas linhas de produtos ou serviços, pequeno capital, baixas margens de lucro, serviços ou produtos inseridos em uma pequena área geográfica e utilização de sistemas de informações manuais. Já EL-NAMAKI (p.80, 1990) observa que a pequena empresa absorve novas tecnologias de informação de forma lenta e limitada nos seus processos administrativos ou de produção, por falta de tecnologia específica.
Nas empresas desenvolvedoras de software (em especial nas pequenas), o padrão de desenvolvimento começa pela programação, isto quando não se resume somente a ela, conforme BELLOQUIM (p.13, 2002). Além disso, percebem-se outros problemas:

- A criatividade do engenheiro de software (essencial em quaisquer etapas no ciclo de vida) é bem vinda, porém sem restrições tende a nunca chegar a um fechamento. Por outro lado, ferramentas disciplinadoras utilizadas sem nenhum contexto criam entraves burocráticos que acabam por sufocar a inovação, conforme Grady Booch, citado por QUATRANI (p.5, 2000).
- O processo é gerenciado de maneira informal, geralmente reduzido ao controle de tempo, seja o período de desenvolvimento, ou o prazo de entrega que, ainda assim, não é cumprido.
- Também encontram-se problemas naquelas empresas que optam por técnicas isoladas de modelagem de software, com o intuito de melhorar a qualidade de seus produtos. A adoção de ferramentas de engenharia de software, neste contexto, torna-se contraproducente na medida em que não estão integradas em um processo maior, de acordo com BELLOQUIM (p.13, 2002). Este problema causa na equipe de desenvolvimento, a impressão de que técnicas de modelagem (e gerenciamento) não funcionam na prática.

As pequenas equipes de desenvolvimento de software têm sido alvo de metodologias que visam garantir a qualidade do produto final (satisfação do cliente). Uma delas é a *Extreme Programming* onde, para BECK e FOWLER (p.5, 2000), trata-se de uma metodologia ágil para equipes pequenas e médias desenvolvendo software com requisitos vagos e em constante mudança. Segundo WUESTEFELD (p.1, 2003) está baseada nas seguintes práticas:

- *The customer is always available*: o usuário final deve incorporar-se à equipe de desenvolvimento em todas as etapas do projeto.
- *Metaphor*: a comunicação entre a equipe sobre os módulos do software é mais produtiva se for realizada através do auxílio de metáforas.

- *Planning the game*: priorizar a produção dos módulos do software que agregam maior valor ao produto.
- *Small releases*: deve-se trabalhar com pequenas versões do software a cada momento, contribuindo assim para a produtividade do cliente e, conseqüentemente, da equipe.
- *Acceptance tests*: testes definidos e implementados pelo cliente.
- *Test first design*: criação da unidade de testes antes do código.
- *Continuous integration*: os módulos devem ser testados isoladamente somente após a aferição de 100% isento de erros, são integrados.
- *Simple design*: procurar desenvolver o software, em todas as suas etapas, o mais simples possível.
- *Refactoring*: tornar o software simples, sempre que necessário.
- *Pair programming*: o software deve ser desenvolvido por duas pessoas, compartilhando as mesmas ferramentas de trabalho.
- *Move people around*: deve haver um revezamento, a cada duas horas, das duplas de programadores.
- *Collective code ownership*: todos os programadores devem conhecer o código e possuir autoridade para alterá-lo a qualquer momento.
- *Coding standards*: padronizar o código.
- *40 hour week*: é contraproducente trabalhar em períodos maiores que uma semana.

De um modo geral, a *Extreme Programming* não trouxe contribuições inéditas expressivas para o processo de desenvolvimento de software na pequena empresa. Na verdade, a aplicabilidade de alguns dos itens observados pode ser inviável (em algumas situações, até mesmo impraticável), como por exemplo, a incorporação *full-time* do usuário final à equipe de desenvolvimento. Outras práticas já vêm sendo realizadas corriqueiramente no cotidiano das empresas, como por exemplo, pequenas versões do software, testes isolados e efetuados (também) pelo usuário final. Algumas preconizam o óbvio (40 horas semanais, software simples). Porém, sua validade reside no fato de resgatar e realizar um *upgrade* nas boas práticas de desenvolvimento de software.

3. ESTUDO DE CASO

Este tópico descreve o processo de implantação do gerenciamento de projetos no processo de produção de uma pequena software-house e os resultados surgidos com esta abordagem. Para verificação, optou-se pelo método do estudo de caso que, de acordo com Goode e Hatt citados por BRESSAN (p.1, 2003), não sendo uma técnica específica, pode ser utilizado como meio para organizar dados preservando o caráter unitário do objeto social estudado, além de descrever situações gerenciais, conforme Bonoma, também citado pelo autor.

O processo de implantação de gerenciamento de projetos teve início em maio de 2000, em uma empresa de fornecimento, instalação e manutenção de hardware, que possuía em sua carteira de clientes, basicamente pequenas e médias empresas, atuando no mercado regional de Cascavel – PR há 8 anos, sendo que passou a oferecer serviços de desenvolvimento de sistemas de informações desde 1998. No período em que iniciou este estudo de caso, o faturamento da empresa era, aproximadamente, R$ 10.000,00 ao mês. A equipe de desenvolvimento de sistemas era composta de um analista de sistemas, três programadores e um analista de suporte e não utilizavam nenhuma ferramenta automatizada de geração de software, bem como ferramentas de documentação de sistemas. Os produtos eram desenvolvidos em um ambiente integrado de desenvolvimento e programação visual, com banco de dados nativo deste ambiente. Os profissionais de desenvolvimento eram alocados conforme a necessidade cotidiana na empresa, sendo que todos possuíam conhecimentos avançados no ambiente de programação, ainda que não conhecessem metodologias formais de desenvolvimento de software (analise, projeto e programação estruturada e orientada a

objetos, por exemplo). Os produtos eram gerenciados informalmente, havendo apenas controle de horário através de cartão-ponto. Como o software era desenvolvido de maneira modular, a medida em que os módulos eram concluídos, os mesmos eram implantados no cliente para que fosse testado em ambiente de produção.
Em função das características acima mencionadas, a empresa tinha enorme dificuldade em contabilizar os lucros (exatos) decorrentes da comercialização de softwares. Da mesma forma, encontrava dificuldades em confeccionar orçamentos, o que geralmente era feito de acordo com o número (estimado) de tabelas (e campos pertencentes a cada tabela) que seriam manipuladas pela aplicação.
O presente estudo de caso propõe-se a avaliar os resultados da utilização do gerenciamento formal de projetos. Para efeitos de aferição de resultados será utilizada uma metodologia multidimensional baseada na satisfação percebida pelos engenheiros de software. Além disso, irá comparar estes resultados com a situação atual da empresa, transcorridos 2 anos após a implantação da gestão de projetos.

3.1 Implantação de desenvolvimento por projetos

O passo decisivo para a implantação de gerenciamento de projetos na empresa estudada foi a constatação, por parte do proprietário, que os softwares desenvolvidos poderiam ter maior competitividade se distintos da concorrência. Tecnicamente, os produtos não apresentavam grandes diferenciais, visíveis ao cliente, em relação aos concorrentes. Portanto a estratégia voltou-se ao cumprimento de prazos, estimativa de custos e menor índice de manutenção após a entrega do software.
Uma vez verificado que os pontos críticos no processo estavam mais próximos do gerenciamento, do que em ferramentas ou metodologias de desenvolvimento, foi estabelecido um programa de treinamento específico em técnicas de gestão de projetos. É importante frisar que o processo de implantação foi discutido e planejado com toda a equipe (proprietário e colaboradores) sem distinção de funções, visto que as tarefas do processo eram (até certo ponto) herméticas e, portanto, optou-se por uma linha de comunicação aberta, entre todos os envolvidos no desenvolvimento do produto, com o objetivo de formar uma visão sistêmica do mesmo. A formatação do programa de treinamento consumiu 8 horas, distribuídas em 4 dias.
A primeira etapa do programa de implantação constituiu-se de uma palestra sobre gerenciamento de projetos, mostrando sua importância no desenvolvimento de produtos, em termos de eficiência (razão entre recursos consumidos e objetivos alcançados) e eficácia (razão entre objetivos alcançados e objetivos atingidos). Estes aspectos foram considerados importantes para introduzir a idéia matemática de mensuração de resultados, no processo de desenvolvimento. Esta etapa consumiu 2 horas do 5º dia, porém foi efetuada em horário extra às atividades de desenvolvimento.
A seguir, ministrou-se mini-cursos específicos sobre ferramentas de gerenciamento de projetos. Adotou-se o modelo PERT/CPM, por julgá-lo suficientemente fácil para aprendizagem e prática. Dividiu-se o modelo adotado em 4 etapas (estrutura de decomposição de atividades, PERT/CPM, PERT/Custo e aceleração de atividades), sendo que cada etapa consumiu 3 horas/dia. Após o curso, a equipe já estava apta a aplicar os conhecimentos diretamente no ambiente de produção.
A próxima etapa foi baseada no *Unified Software Development Process* e na proposta descrita por BEZERRA (p.78, 2001), a qual mostrava a aplicabilidade, as possibilidades e as vantagens do planejamento e (posterior) execução simultânea de processos de desenvolvimento de software, bem como a utilização do modelo PERT/CPM nesta abordagem. Esta etapa foi concluída após 6 horas de treinamento, dividida em 2 dias.
Finalmente, utilizaram-se algumas das boas práticas da atividade de programação, atualizadas pela disciplina da *Extreme Programming*, para concluir o programa de treinamento. Foram enfatizados os aspectos referentes à importância da participação do usuário final na definição

e implementação de testes, programação modular, desenvolvimento dos módulos que mais agregam valor final ao produto em primeiro lugar e, padronização de código (bem como demais subprodutos decorrentes de cada interação no processo de desenvolvimento). Esta etapa consumiu 4 horas em 1 dia de trabalho.

Ao todo, o programa de treinamento levou aproximadamente 32 horas, distribuídos em 12 dias. É importante salientar que a equipe de desenvolvimento contava com o acompanhamento do instrutor em período integral. Outros fatores tornaram-se facilitadores do processo:

- A estrutura organizacional existente poderia ser enquadrada, ainda que de maneira incipiente, na categoria por projetos, conforme VALERIANO (p.85, 1998), onde toda a empresa estava voltada a atender os interesses de cada projeto. Ou seja, já tinham um certo foco na execução de projetos, apesar de informalmente gerenciados.
- Não havia nenhum outro projeto em andamento. O projeto escolhido como piloto teve início justamente após o início do programa de treinamento. Além disso, não tratava de um sistema complexo.
- Toda a equipe (analistas, programadores e gerente de projeto) recebeu treinamento das mesmas ferramentas e técnicas, sem distinção de função. Isto foi um fator importante por agregar a equipe em torno de um único objetivo (sem a competitividade existente entre as diversas funções) que era o cumprimento de etapas pré-determinadas. Além disso, todos tiveram o conhecimento sobre como o processo seria medido e, conseqüentemente, controlado.

O projeto piloto, gerenciado formalmente, consistiu de um sistema de informação comercial, contendo os módulos clientes e fornecedores, compras, vendas e estoque, contas a receber, contas a pagar e fluxo de caixa para uma indústria de insumos agrícolas, atuando no mercado regional de Cascavel – PR desde 1984, cujo faturamento no ano de 1999 foi de aproximadamente R$ 1.000.000,00. Esta indústria, na época, contava com 20 funcionários e 3 gerentes (financeiro, produção e administrativo / comercial) e não tinha experiência anterior na utilização de sistemas informatizados de gestão e operações, porém devido ao aumento expressivo de faturamento entre os anos de 1998 e 1999, o proprietário resolveu utilizar ferramentas informatizadas como apoio ao processo decisório.

3.2 Instrumento de validação

O instrumento utilizado para medir o impacto da implantação da metodologia de desenvolvimento por projetos, na empresa, foi desenvolvido por Torkzadeh e Doll em 1999 (MAÇADA, BORESTEIN e MORALES *et al*, p.3, 2000) e fundamenta-se em 4 dimensões: produtividade, inovação, satisfação e controle. Optou-se por este instrumento devido suas características de avaliação mutidimensional do impacto sobre os indivíduos, facilidade de aplicação e sua capacidade em ser adaptado para diversos contextos. O modelo proposto por Torkzadeh e Doll, constitui-se da utilização da pesquisa *survey*, sob a forma de um questionário, para determinar quantitativamente (em uma escala Likert, onde nada = 1, um pouco = 2, moderadamente = 3, muito = 4 e muitíssimo = 5) características de constructos das 4 dimensões representativas do impacto da adoção de determinada tecnologia na satisfação do usuário final. As dimensões, os constructos e as questões, do instrumento implementado por Torkzadeh e Doll, podem ser visualizadas no QUADRO 1:

QUADRO 1 – Instrumento de validação

Dimensão	Constructo	Questões
Produtividade	Em que medida a aplicação interfere na produção do usuário em determinada unidade de tempo	a) O sistema poupa-me tempo b) O sistema melhora minha produtividade c) O sistema permite-me melhores resultados do que seria possível executar sem ele
Inovação	Em que medida a aplicação ajuda a	d) O sistema ajuda-me a criar novas idéias

	criar ou tentar expressar novas idéias em seu trabalho	e) O sistema permite-me propor novas idéias f) O sistema coloca-me diante de idéias inovadoras
Satisfação do Usuário	Em que medida a aplicação ajuda o usuário a criar valor para os clientes internos e externos à organização	g) O sistema melhora o serviço do usuário h) O sistema melhora a satisfação do usuário i) O sistema vai ao encontro às necessidades do usuário
Controle Gerencial	Em que medida a aplicação ajuda a regular processos e desempenho	j) O sistema ajuda no controle gerencial do processo de trabalho k) O sistema melhora o controle do gerenciamento l) O sistema ajuda no controle do gerenciamento de performance do processo de trabalho

Fonte: adaptado de BEZERRA (p.84, 2001)

3.2.1 Resultados imediatos

Após o desenvolvimento do projeto piloto, o qual foi implantado em 11 semanas, aplicou-se o questionário aos envolvidos no projeto (desenvolvedores e usuários finais). Este questionário foi aplicado individualmente, por escrito, sem identificação (com o intuito de minimizar tendências ou influências do grupo e manter a isenção). Após a tabulação dos dados, o grupo foi reunido para discutir os resultados. A análise da satisfação dos usuários finais (clientes), em termos quantitativos pode ser consultada em BEZERRA (p.92, 2001). Com relação à satisfação da equipe de desenvolvimento (objeto deste estudo), os resultados podem ser observados na FIGURA 1:

FIGURA 1: Resultados obtidos

	Escala Likert **Σ obtido** **%** **aprovação**
1 **2** **3** **4** **5**	
Produtividade	
a) Poupa tempo	
	0 0 1 2 2 **21** **84,00**
b) Melhora a produtividade	
	0 1 1 1 2 **19** **76,00**
c) Melhores resultados	
	1 1 2 1 0 **13** **52,00**

Totais	
	1
	2
	4
	4
	4
	53
	70,667
Inovação	
d) Criar novas idéias	
	0
	1
	1
	1
	2
	19
	76,00
e) Propor novas idéias	
	0
	0
	1
	2
	2
	21
	84,00
f) Diante de inovação	
	0
	1
	1
	1
	2
	19
	76,00
Totais	
	0
	2
	3
	4
	6
	59
	78,667
Satisfação	
g) Melhora o serviço	
	0
	1
	2
	2
	0
	16
	64,00
h) Melhora a satisfação	
	1
	0
	1
	2
	1
	17
	68,00

i) Encontro às necessidades	
	0
	1
	1
	1
	2
	19
	76,00
Totais	
	1
	2
	4
	5
	3
	52
	69,333
Controle	
j) Processo do trabalho	
	0
	1
	2
	1
	1
	17
	68,00
k) Controle do gerenciamento	
	0
	1
	1
	2
	1
	18
	72,00
l) Performance e trabalho	
	0
	1
	1
	1
	2
	19
	76,00
Totais	
	0
	3
	4
	4
	4
	54
	72,00

Fonte: adaptado de BEZERRA (p.111, 2001)

Os resultados obtidos mostram que o desenvolvimento do projeto piloto, gerenciado formalmente, trouxe índices significativos de satisfação para a equipe de desenvolvimento. Os números apresentados, no presente estudo, merecem algumas considerações:

- O constructo "inovação" teve o maior percentual de aprovação (78,667%). Este aspecto é facilmente explicado, visto que o processo de gestão desenvolvimento era bastante amador. A inovação percebida reside-se no fato de que o gerenciamento pôde medir

matematicamente o progresso do projeto – circunstância inédita, até então, para a equipe de desenvolvimento. Com relação à "inovação", o quesito "o sistema permite-me propor novas idéias" surpreendeu pelo elevado índice de aprovação (84%). O motivo alegado pela equipe foi a possibilidade de compartilhar idéias, promovido pelos aspectos interativos e incrementais da *Unified Software Development Process* e de uma abordagem de engenharia simultânea no processo de desenvolvimento de software.

- O constructo "controle" apresentou o segundo maior percentual de aprovação (72%), sendo que o quesito "o sistema ajuda no controle do gerenciamento de performance do processo de trabalho" contribuiu com 76%, o que não causou surpresa, bem como os quesitos "o sistema melhora minha produtividade" do constructo "produtividade" e "o sistema vai ao encontro às necessidades do usuário" do constructo "satisfação".
- O quesito "o sistema permite-me melhores resultados do que seria possível executar sem ele" do constructo "produtividade" apresentou o menor índice de aprovação: pouco mais da metade da equipe de desenvolvedores confirmou este aspecto. Este fato é explicado por HEHN (p.41, 1999), ao analisar as resistências às mudanças intrínsecas às pessoas ao saírem, ou serem obrigadas a sair, da zona de conforto onde, até então, julgavam ser absolutas conhecedoras de suas atividades e, portanto, não necessitavam serem gerenciadas. Outra razão para o resultado apresentado é a constatação de que o software, independente de técnica, metodologia de desenvolvimento, ferramenta e/ou gerenciamento do processo, irá ser concluído, não considerando fatores custo, prazo de entrega e funcionalidade. Além disso, existe a crença (errada) de que os problemas decorrentes do desenvolvimento informal (custos e prazos estourados e não funcionalidade) fazem parte do processo e que, sendo assim, não existem razões para minimizar esta situação.

De um modo geral, com relação à implantação da gestão formal de projetos, alguns aspectos foram observados:

- As etapas de planejamento de projeto e as etapas do ciclo de desenvolvimento de software, principalmente levantamento de requisitos, análise e projeto, levaram um tempo maior do que aquele existente anteriormente em sistemas da mesma natureza, o que levou a equipe a ter, preliminarmente, uma idéia de baixa produtividade, uma vez que o software, propriamente dito, ainda não havia sido iniciado. Porém, após este momento, a etapa de codificação mostrou-se extremamente rápida e "mecânica", conforme já observado por PRESSMAN (p.35, 2001).
- Verificaram-se dificuldades na adoção de um ambiente (e metodologia) de desenvolvimento que seja totalmente integrado. A princípio, cogitou-se a utilização total da orientação a objetos em todos os aspectos do projeto, porém encontraram-se barreiras na aquisição e domínio de um banco de dados orientado a objetos, por falta de literatura específica, opções de custo acessível no mercado e treinamento.
- Infelizmente não se adotou nenhuma métrica científica para medição do software, sendo que este valor foi estimado de acordo com a experiência do grupo de trabalho.
- A equipe de desenvolvedores mostrou-se arredia na primeira etapa (palestra sobre gerenciamento de projetos), sentindo-se "vigiada", "como se não estivessem executando suas tarefas adequadamente". A partir da definição, em conjunto do programa de treinamento, a equipe passou a mostrar-se mais colaborativa. Este aspecto, segundo eles, tornou-se crucial para o sucesso da utilização do gerenciamento de projetos, formalizando a idéia de que todos (e todas as funções envolvidas) fazem parte de um processo maior, que é a satisfação do cliente.

Para que a implantação do gerenciamento de projetos, neste estudo de caso, obtivesse sucesso, foi necessária a quebra de alguns paradigmas, já observados por HEHN (p.80-86, 1999). O primeiro deles foi a busca por resultados imediatos (processos de transformação levam mais

tempo para mostrar resultados). Administraram-se expectativas sem gerar ansiedade e resultados irreais. Evitou-se também a indução ao erro com um canal aberto e efetivo de comunicação entre proprietário da software-house, consultor, desenvolvedores e usuários finais, de forma a tornar claro o significado de todas as ações, os pontos de controle e os objetivos esperados.

3.2.2 Resultados posteriores

Após a análise dos resultados, uma grande dúvida persistiu: projetos posteriores seriam gerenciados formalmente pela empresa? Se sim, que resultados poderiam ser percebidos? Se não, qual o motivo que levou a desistência de se utilizar esta abordagem? A partir destas questões, passados 2 anos após o estudo, procurou-se avaliar a software-house novamente, utilizando-se a mesma ferramenta, com o intuito de verificar se o gerenciamento de projeto, naquele contexto, tratou-se de uma metodologia passageira, que não representou nenhum avanço, ou um *change trigger* (avanço tecnológico provocador de rupturas), conforme HEHN (p.31, 1999).

A empresa, no contexto em que efetuado o novo estudo, contava com o mesmo número de desenvolvedores, apesar de terem substituído 2 colaboradores no ano de 2001. Quando, da primeira avaliação, aproximadamente 75% de seu faturamento correspondia à venda, manutenção e instalação de hardware, sendo que transcorridos 2 anos, a comercialização e manutenção de sistemas de informação já representava até 50%, em média, do faturamento total, que atingia até R$ 25.000,00 ao mês. Verificou-se, nesta ocasião, a utilização maciça da gestão de projetos, não somente em desenvolvimento de software, mas também na forma como a empresa passou a comercializar, nas palavras do proprietário, "soluções e não mais produtos". Neste cenário, aplicou-se novamente o questionário e seus resultados podem ser visualizados na FIGURA 2:

FIGURA 2: Resultados atualizados em 2002

	Escala Likert **Σ obtido** **%** **aprovação**
1 **2** **3** **4** **5**	
Produtividade	
a) Poupa tempo	
	0 0 1 2 2 **21** **84**
b) Melhora a produtividade	
	0 0 1 2 2 **21** **84**

c) Melhores resultados	
	0
	0
	1
	1
	3
	22
	84
Totais	
	0
	0
	3
	5
	7
	64
	85,33
Inovação	
d) Criar novas idéias	
	0
	1
	1
	1
	2
	19
	76
e) Propor novas idéias	
	0
	1
	1
	2
	1
	18
	72
f) Diante de inovação	
	0
	1
	1
	2
	1
	18
	72
Totais	
	0
	3
	3
	5
	4
	55
	73,33
Satisfação	
g) Melhora o serviço	
	0
	0
	1
	2
	2
	21
	84

h) Melhora a satisfação	
	0
	1
	1
	2
	1
	18
	72
i) Encontro às necessidades	
	0
	0
	1
	2
	2
	21
	84
Totais	
	0
	1
	3
	6
	5
	60
	80
Controle	
j) Processo do trabalho	
	0
	0
	1
	2
	2
	21
	84
k) Controle do gerenciamento	
	0
	0
	0
	2
	3
	23
	92
l) Performance e trabalho	
	0
	0
	1
	2
	2
	21
	84
Totais	
	0
	0
	2
	6
	7
	65
	86,67

Em um primeiro momento, pode-se constatar o aumento expressivo na satisfação observada pela equipe de desenvolvimento de software, principalmente nos constructos "controle", que passou de um percentual de 72% para 86,67% de aprovação, "produtividade", de 70,667% para 85,33%, e "satisfação", de 69,333% para 80%. Infelizmente, a empresa procurou agregar poucas técnicas e metodologias no desenvolvimento do processo de engenharia de software, o que explica a queda (de 78,667% para 73,33%) no constructo "inovação". Os processos que levam à qualidade possuem a característica da "melhoria contínua" e a estagnação (ou acomodação), neste contexto, contribui para a percepção de não estar praticando a inovação.
Desta forma, de acordo com o modelo CMM, atualmente a empresa encontra-se bastante próxima ao nível 2 – repetitivo, devido às características de controle sobre documentação, prazos, custos e funcionalidade. A afirmação de FIORINI, STAA e BAPTISTA (p. 33, 1998) – "as organizações podem usar de maneira vantajosa processos descritos em níveis de maturidade superiores ao que se encontram" – mostrou-se verdadeira neste caso. A empresa trabalha no sentido de coordenar todos os grupos de trabalho (administrativo / financeiro, desenvolvimento, comercial), foi adotada uma ferramenta de desenvolvimento padrão, assim como caminham para a adoção de medidas de desempenho e métricas de software (características encontradas no nível 3 – definido – do CMM).

3.3 Limitações e recomendações ao estudo

O presente estudo focalizou a gestão de projetos de software no que diz respeito somente à engenharia do processo de desenvolvimento de software. Análises mais completas deveriam ser efetuadas no sentido de avaliar o impacto organizacional como um todo, principalmente em aspectos administrativos e financeiros, recursos humanos (gerência e liderança, equipe de trabalho, motivação e administração de conflitos), gestão de documentação técnica, e de riscos.
A implantação da gestão de projetos na empresa analisada foi relativamente fácil devido à sua característica de já trabalhar com equipes alocadas a cada projeto individual. Desta forma, recomenda-se estudos dos impactos e dos resultados obtidos naquelas empresas que possuem uma estrutura organizacional já formalizada, levando em consideração o surgimento de conflitos devido à quebra de modelos mentais e organizacionais.
Apesar do desenvolvimento de software estar caminhando muito rapidamente para um processo real de engenharia, inclusive com a componentização de elementos, ainda hoje se vale muito da capacidade individual do desenvolvedor. Neste sentido, nas software-house's onde os colaboradores estejam envolvidos em mais de um projeto, uma grande dificuldade será a mensuração do tempo de alocação de cada um a cada tarefa de cada projeto. Talvez o principal desafio seja balancear a criatividade com uma maneira disciplinada de controle da atividade de desenvolvimento.

4. CONCLUSÕES

A abordagem de desenvolvimento por projetos na pequena empresa encontra algumas barreiras de caráter empírico, por exemplo: é comum adotar a postura de que um bom programador será um bom analista de sistemas e que, um bom analista de sistemas será um bom gerente de projetos. Esta situação até pode ocorrer (e quando acontece, é bastante saudável), mas não como via de regra. Este é um paradigma a ser quebrado em desenvolvimento de software por projetos. As competências devem ser alocadas conforme a atividade a ser desenvolvida. Isto é importante no sentido de determinar uma pessoa plenamente apta a gerenciar o projeto, no sentido literal da palavra "gerente" que, de acordo com VALERIANO (p.149, 1998), deve possuir conhecimentos organizacionais e técnicos, habilidades de comando e relacionamento pessoal, atitudes de disciplina, interesse por questões administrativas, entrosamento com pessoal externo à organização e ser um estrategista.

Administrativamente verifica-se que o ideal seria possuir uma estrutura organizacional com grande foco no gerenciamento de projetos, porém interagindo fortemente com os demais aspectos da empresa. Neste ponto, há que se concordar com Watts Humphrey, citado por TEIXEIRA JÚNIOR e SANCHES (p.1, 2001), quando afirma que "a ausência de práticas administrativas é a principal causa de sérios problemas enfrentados pelas organizações: atraso em cronogramas, custo maior do que o esperado e presença de defeitos, ocasionando uma série de inconveniências para os usuários e enorme perda de tempo e de recursos dos desenvolvedores". Em uma pequena software-house, algumas pessoas seriam alocadas ao projeto, enquanto outras estariam dedicadas ao suporte e manutenção dos sistemas já implantados, bem como colocando no mesmo nível hierárquico, gerentes de projeto e gerentes funcionais caracterizando uma estrutura matricial balanceada, conforme VALERIANO (p.85, 1998).

Evidentemente, a empresa que adota a postura do gerenciamento de projetos, deve estar em um nível de maturidade organizacional elevado ou, no mínimo, estar consciente desta necessidade. Isto é determinante pelo fato de que problemas irão existir, porém serão mais facilmente equacionados e resolvidos. Outra postura que irá contribuir positivamente é a consciência de que a gestão formal de projetos é apenas uma das etapas para o objetivo maior da empresa, que é a qualidade total e, neste sentido, é importante observar que deverá ocorrer uma evolução para técnicas, metodologias e modelos mais específicos.

Finalmente, verifica-se a validade do modelo CMM que, em seus níveis iniciais, não se detém em especificidades técnicas, além do aspecto gerenciamento de projeto. A partir do momento em que o processo se torna repetitivo e, portanto, mais facilmente controlável, ocorre a preocupação com características mais especializadas do desenvolvimento do software, onde caberiam modelos como PSP (*Personal Software Process*), TSP (*Team Software Process*), além do estabelecimento de métricas (no nível 3 – definido), e aspectos de gerenciamento da qualidade (níveis 4 – gerenciado e 5 – em otimização). Tal modelo mostrou-se perfeitamente adaptável a uma pequena empresa desenvolvedora, com resultados bastante satisfatórios em termos de produtividade e qualidade de processo, enquanto observado pelos engenheiros de software.

REFERÊNCIAS BIBLIOGRÁFICAS

1. BECK, Kent; FOWLER, Martin. **Planning extreme programming**. New York: Addison-Wesley, 2000.
2. BELLOQUIM, Átila. Modelagem de software: ontem, hoje e amanhã. **Developers' Magazine**. Rio de Janeiro: Axcel Books, ano 6, n.70, p.10-13, jun.2002.
3. BEZERRA, Cicero Aparecido. **Projeto de sistemas de informação baseado em qualidade**: uma abordagem voltada à pequena empresa. Florianópolis, 2001. Dissertação (Mestrado em Engenharia de Produção e Sistemas) – Programa de Pós-Graduação em Engenharia de Produção e Sistemas, UFSC, 2001.
4. BRESSAN, Flávio. **O método do estudo de caso**. Disponível em <http://www.fecap.br/adm_online/art11/flavio.htm> Acesso em: 07/01/2003.
5. CASAROTTO FILHO, Nelson; FÁVERO, José Severino; CASTRO, João E. E. **Gerência de projetos / Engenharia simultânea**. São Paulo: Atlas, 1999.
6. EL-NAMAKI, M. S. S. Small Business: the myths and the reality. **Long range planning**. Great Britain, v.23, n.4, p.78-87, 1990.
7. FIORINI, Soeli T; STAA Arndt von; BAPTISTA, Renan Martins. **Engenharia de software com CMM**. Rio de Janeiro: Brasport, 1998.
8. HARRIS, David. **Systems analysis and design for the small enterprise**. 2. ed. Orlando: The Dryden Press, 1999.
9. HEHN, Herman F. **Peopleware**: como trabalhar o fator humano nas implementações de sistemas integrados de informação (ERP). São Paulo: Editora Gente, 1999.

10. MAÇADA, Antonio C. G; BORENSTEIN, Denis; MORALES, Bayardo *et al*. **Medindo a satisfação dos usuários de um sistema de apoio à decisão**. Florianópolis, 2000. Anais do 24º Encontro da Associação Nacional dos Programas de Pós-Graduação em Administração. Rio de Janeiro: ANPAD, 2000. CD-ROM.
11. MARTINS, José Carlos Cordeiro. **Gestão de projetos de desenvolvimento de software**. Rio de Janeiro: Brasport, 2002.
12. PAULA FILHO, Wilson de Pádua. **Engenharia de software**: fundamentos, métodos e padrões. Rio de Janeiro: LTC, 2001.
13. PETERS, James F; PEDRYCZ, Witold. **Software engineering**: an engineering approach. New York: John Wiley & Sons, 2000.
14. PETRINI, Maira; POZZEBON, Marlei. **Interação usuário-sistema**: um estudo empírico sobre a proatividade no uso de sistemas de informação. Florianópolis, 2000. Anais do 24º Encontro da Associação Nacional dos Programas de Pós-Graduação em Administração. Rio de Janeiro: ANPAD, 2000.
15. PRESSMAN, Roger. **Software engineering**: a practitioner's aproach. 5. ed. New York: McGraw-Hill, 2001.
16. QUATRANI, Terry. **Visual modeling with Rational Rose and UML**. 2. ed. New York: Addison-Wesley, 2000.
17. STANDISH GROUP, The. **Chaos chronicles III**. West Yarmouth, 2003. 400 p. Relatório técnico.
18. TEIXEIRA JÚNIOR, Waine; SANCHES, Rosely. **Proposta de um modelo de processo de planejamento de projeto de software para melhoria de gerenciamento de projetos**. Curitiba, 2001. Anais do XII Congresso Internacional de Tecnologia de Software. Curitiba: CITS, 2001. CD-ROM.
19. VALERIANO, Dalton L. **Gerência em projetos**: pesquisa, desenvolvimento e engenharia. São Paulo: Makron Books, 1998.
20. WUESTEFELD, Klaus. **Xispê**: Extreme programming. Disponível em <http://www.xispe.com.br> Acesso em: 07/01/2003.