"QUAL A IMPORTÂNCIA DA ADOÇÃO DA NORMA ISO 12207 NAS EMPRESAS DE DESENVOLVIMENTO DE SOFTWARE?".

MARCELO NOGUEIRA
Universidade Paulista
Rua Dr. Bacelar 1212 – 4º - CEP 04026-002 – São Paulo – SP.
marcelo@noginfo.com.br

Resumo

As empresas de desenvolvimento de sistemas têm seus recursos materiais e humanos tanto quanto sobrecarregados, diante da demanda a ela submetida. Porém a não adoção de uma norma de qualidade de software, ou por falta de cultura ou até por desconhecimento da existência da norma, levam projetos de suma relevância ao insucesso e aumentando ainda mais a quantidade de casos dos sistemas inacabados, conseqüentemente desperdiçando tempo, recursos financeiros e não atendendo as necessidades dos clientes e nem do desenvolvedor.

Palavras-chave: Qualidade de Software, ISO 12207, Engenharia de Software.

Abstract

The systems development companies have their material and human resources as much as overloaded, in front of the demand to her submitted. However the not adoption of a software quality norm, or for lack of culture or until for ignorance of the existence of the norm, carry projects of vanishes relevance to the failure and even increasing the quantity of cases of the unfinished systems, consequently wasting time, financial resources and not attending the needs to clients and neither of the developer.

Key words: Software quality, ISO 12207, Software Engineering.

1. INTRODUÇÃO

Num ambiente competitivo e de mudança cada vez mais complexo, a gestão adequada da Informação assume uma importância decisiva no processo de tomada de decisão nas organizações.

Tratando-se de um tema simultaneamente abrangente e especializado, a adoção da Engenharia de Software como linha base da Gestão da Informação, possibilitará, não só desenvolver e consolidar os conhecimentos no desenvolvimento de software para encarar com confiança os novos desafios no mundo dos negócios, e também reforçar as competências profissionais, mantendo-se atualizado em relação ao potencial dos sistemas de informação e das novas tecnologias numa perspectiva empresarial e competitiva globalmente.

A partir do conhecimento adquirido de normas de Qualidade de Software, o desenvolvedor será elemento multiplicador de soluções, contribuindo e agregando valor aos sistemas novos e para os já existentes, com aplicação de metodologias e tecnologias adequadas, capazes de gerir com sucesso as informações relevantes aos negócios aplicáveis, trazendo ás organizações, vantagens competitivas.

No estudo da Engenharia de Software, o autor Roger S. Pressman [PRESSMAN02], demonstra preocupação com a "Crise do Software" que atualmente ele intitula como "Aflição Crônica", chegando a determinar números expressivos sobre a não finalização de projetos de sistemas começados e não terminados.
Num mundo cada vez mais de recursos financeiros escassos, como é possível aceitar tal desperdício de tempo e dinheiro. O mesmo autor também aponta para o possível problema causador de tal

absurdo: “A falta de adoção de métodos, ferramentas e procedimentos no desenvolvimento de software e a difícil relação de entendimento entre o usuário com o desenvolvedor”.

Várias técnicas de modelagem foram criadas e o desenvolvedor mesmo assim ainda tem dificuldades de realizar um projeto de sistemas livre de manutenções e re-trabalhos, condenando diretamente a qualidade do produto.

A adoção da norma ISO/IEC 12207:1997, pelo desenvolvedor de software, direciona como estruturar e gerenciar o ciclo de desenvolvimento, proporcionando acompanhamento de todo o processo, permitindo que o software venha representar a realidade da empresa modelada para geração de um sistema customizado, atendendo assim seus requisitos e as necessidades desta empresa.

Adotar a norma permite padronizar os processos de desenvolvimento de software, aumentando a aderência a uma ferramenta de apoio a modelagem do software agregando a elas facilidades de relacionamento e compreensão do Usuário e do Desenvolvedor, utilizando métodos sistêmicos que diminuirão erros na modelagem, e por conseqüência a manutenção exagerada, eliminação de custos indevidos, extinção do tempo desnecessário dedicado ao re-trabalho, obtendo qualidade no software, bem como criar a real possibilidade de extrair de um sistema, informações relevantes que venham não só para contribuir com a decisão, mas para ser um fator de excelência empresarial, permitindo novos negócios, permanência e sobrevivência num mercado atuante.

2. RELEVÂNCIA

Atualmente com a visão global permitindo a participação nas exportações de software para outros países, cada vez mais a qualidade no processo de desenvolvimento e do produto de software ganha maior observação e adoção das melhores práticas e soluções tecnológicas.

3. ENGENHARIA DE SOFTWARE

Segundo Rezende [REZENDE99], Engenharia é a arte das construções, embasada no conhecimento científico e empírico, adequada ao atendimento das necessidades humanas.

Engenharia de Software é a metodologia de desenvolvimento e manutenção de sistemas modulares, com as seguintes características [REZENDE99]:

- Adequação aos requisitos funcionais do negócio do cliente e seus respectivos procedimentos pertinentes;
- Efetivação de padrões de qualidade e produtividade em suas atividades e produtos;
- Fundamentação na tecnologia da informação disponível, viável e oportuna;
- Planejamento e gestão de atividades, recursos, custos e datas.

Segundo Pressman [PRESSMAN95], Engenharia de Software é:

- “O estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um software que seja confiável e que funcione eficientemente em máquinas reais”;
- “Descendente da engenharia de sistemas e de hardware. Abrange um conjunto de 3 elementos fundamentais (métodos, ferramentas e procedimentos), que possibilita, ao gerente, o controle do processo de desenvolvimento do software e oferece ao profissional uma base para a construção de software de alta qualidade”.

Segundo Martin [MARTIN91], Engenharia de Software é:

- "É o estudo dos princípios e sua aplicação no desenvolvimento e manutenção de sistemas de software";
- "Tanto a engenharia de software como as técnicas estruturadas são coleções de metodologias de software e ferramentas";

Como conclusão, pode-se relatar que engenharia de software é um conjunto de práticas para desenvolvimento de soluções de software, ou seja, roteiro que pode utilizar diversas técnicas.

A seqüência de passos preestabelecidos permite optar e variar de técnicas e ferramentas na suas diversas fases [REZENDE99].

4. OBJETIVOS DA ENGENHARIA DE SOFTWARE

De um modo geral, considera-se que os objetivos primários da Engenharia de Software são o aprimoramento da qualidade dos produtos de software e o aumento da produtividade dos engenheiros de software, além do atendimento aos requisitos de eficácia e eficiência, ou seja, efetividade [MAFFEO92].

A Engenharia de Software visa sistematizar a produção, a manutenção, a evolução e a recuperação de produtos intensivos de software, de modo que ocorra dentro de prazos e custos estimados, com progresso controlado e utilizando princípios, métodos, tecnologia e processos em contínuo aprimoramento. Os produtos desenvolvidos e mantidos, seguindo um processo efetivo e segundo preceitos da Engenharia de Software asseguram, por construção, qualidade satisfatória, apoiando adequadamente os seus usuários na realização de suas tarefas, operam satisfatória e economicamente em ambientes reais e podem evoluir continuamente, adaptando-se a um mundo em constante evolução [FIORINI98].

Associando a esses objetivos, o termo engenharia pretende indicar que o desenvolvimento de software deve submeter-se a leis similares às que governam a manufatura de produtos industriais em engenharias tradicionais, pois ambos são metodológicos [MAFFEO92].

Com base nos objetivos da Engenharia de Software, fica evidente a necessidade da adoção de uma norma de qualidade para padronizar e gerenciar os processos de desenvolvimento de software.

5. CRISE DO SOFTWARE

Para generalizar o termo, ocorre quando o software não satisfaz seus envolvidos, sejam clientes e/ou usuários, desenvolvedores ou empresa [REZENDE99].

A expressão "Crise do Software", que começou a ser utilizada na década de 60, tem historicamente aludido a um conjunto de problemas recorrentemente enfrentados no processo de desenvolvimento (Construção, implantação e manutenção) de software [MAFFEO92].

Esses problemas não se referem apenas a programas que não funcionam.Na verdade, a chamada "Crise do Software" abrange todos os problemas relacionados a [REZENDE99]:

- Como sistemas computacionais são construídos;

- Como sistemas computacionais são implantados, referindo-se aqui ao processo de substituir sistemas antigos, desativando sistemas correntemente em operação, ou ao processo de instalar um sistema inteiramente novo;
- Como é provida a manutenção da quantidade crescente de software construído, associado a sistemas computacionais cada vez mais complexos;
- Como fazer face à crescente demanda para construção de software, visando satisfazer ao conjunto enormemente variado de anseios por informatização, atualmente detectado na sociedade moderna;
- Como administrar as questões comportamentais, envolvendo os clientes e/ou usuários e a política, cultura e filosofia empresarial.

Apesar da enorme variedade de problemas que caracterizam a crise do software, engenheiros de software e gerentes de projetos para desenvolvimento de sistemas computacionais tendem a concentrar suas preocupações no seguinte aspecto: ”A enorme imprecisão das estimativas de cronogramas e de custos de desenvolvimento” (Tabelas 1, 2, 3 e 4) [LEE02].

Tabela 1- Custos em projeto de software por fase de desenvolvimento [LEE02].

Custos em projeto de software por Fase de Desenvolvimento	
Etapa de Trabalho	**%**
Análise de Requisitos	3
Desenho	8
Programação	7
Testes	15
Manutenção	67

Tabela 2- Custos para correção de erros de software [LEE02].

Custos para Correção de Erros de Software				
Fase de desenvolvimento do software	***% de Desvios*** ($)	***Erros Introduzidos*** (%)	***Erros encontrados*** (%)	***Custo Relativo para*** **Correção**
Análise de Requisitos	5	55	18	1,0
Desenho	25	30	10	1,0 - 1,5
Teste do Código e da Unidade	10			
Teste de Integração	50	10	50	1,0 - 5,0
Validação e Documentação	10			
Manutenção Operacional		5	22	10 – 100

Tabela 3- Excedentes de Custo [LEE02].

Excedentes de Custo	
% Excedente de Custo	**% de Respostas**
<20%	15,5%
21% - 50%	31,5%
51% - 100%	29,6%
101% - 200%	10,2%
201% - 400%	8,8%
>400%	4,4%

Tabela 4- Excedente de Prazo [LEE02].

Excedente de Prazo	
% Excedente de Prazo	**% de Respostas**
<20%	13,9%
21% - 50%	18,3%
51% - 100%	20,0%
101% - 200%	35,5%
201% - 400%	11,2%
>400%	1,1%

6. ANTICRISE DO SOFTWARE

Segundo Rezende [REZENDE99], pode-se resumir que a anticrise é a união e trabalho conjunto e harmonioso de três elementos: Empresa (Alta Administração), Cliente e/ou usuário e a unidade de informática (Desenvolvedores de soluções).

E na prática, cabe principalmente à unidade de informática aceitar este conceito e fazer o possível para a efetivação desta tese (Anticrise), utilizando-se de todos os recursos disponíveis para tal.

A Unidade de informática é um dos principais agentes de mudança nas organizações, preocupando-se com o negócio empresarial, auxiliando efetivamente os gestores nos processos de tomada de decisão, tanto operacionais, como gerenciais e estratégicas.

7. QUALIDADE DE SOFTWARE

Atingir um alto nível de qualidade de produto ou serviço é o objetivo da maioria das organizações. Atualmente não é mais aceitável entregar produtos com baixa qualidade e reparar os problemas e as deficiências depois que os produtos foram entregues ao cliente. [SOMMERVILLE03]

Segundo Machado [MACHADO01], para muitos engenheiros de software, a qualidade do processo de software é tão importante quanto à qualidade do produto. Assim na década de 90 houve uma grande preocupação com a modelagem e melhorias no processo de software. Abordagens importantes como as normas ISO 9000 e a ISO / IEC 12207, o modelo CMM (Capability Maturity Model) e o SPICE (Software Process Improvement and Capability dEtermination) sugerem que melhorando o processo de software, podemos melhorar a qualidade dos produtos.

A qualidade é conseqüência dos processos, das pessoas e da tecnologia. A relação entre e qualidade do produto e cada um desses fatores é complexa. Por isso, é muito mais difícil controlar o grau de qualidade do produto do que controlar os requisitos.[PÁDUA03]

Prevê-se que na primeira década dos anos 2000, após ajustarem seus processos para a produção de software de qualidade dentro de prazos e orçamentos confiáveis, as organizações serão pressionadas por seus concorrentes a reduzir substancialmente os prazos para a entrega de produtos.Organizações que sejam capazes de integrar, harmonizar e acelerar seus processos de desenvolvimento e manutenção de software terão a primazia do mercado [MACHADO01].

Segundo o Ministério da Ciência e Tecnologia [MCT02], ainda que divulgadas na década de 90, o conhecimento e utilização das normas e modelos para qualidade de software, estão distantes de tornar-se realidade nas empresas desenvolvedoras de software, conforme (Tabelas 5, 6, 7 e 8).

Tabela 5- Conhecimento da Norma ISO/IEC 12207 [MCT02].

Conhecimento da Norma NBR ISO/IEC 12207 - Processos de Ciclo de Vida de Software -					
Categorias	**Total**	**Micro**	**Pequena**	**Média**	**Grande**
Conhece e usa sistematicamente	3,9	1,4	3,8	2,6	9,1
Conhece e começa a usar	8,3	6,1	6,8	7,7	14,8
Conhece, mas não usa	55,1	48,6	51,1	64,1	67,0
Não conhece	32,7	43,9	38,3	25,6	9,1

Tabela 6- Conhecimento da Norma ISO 9000 [MCT02].

Conhecimento das Normas ISO 9000 - Gestão da Qualidade					
Categorias	**Total**	**Micro**	**Pequena**	**Média**	**Grande**
Conhece e usa sistematicamente	19,4	3,4	16,3	34,1	44,0
Conhece e começa a usar	14,8	14,1	18,5	12,2	12,1
Conhece, mas não usa	52,4	62,4	50,4	43,9	41,8
Não conhece	13,4	20,1	14,8	9,8	2,2

Tabela 7- Conhecimento do Modelo CMM [MCT02].

Conhecimento do modelo CMM - Capability Maturity Model					
Categorias	**Total**	**Micro**	**Pequena**	**Média**	**Grande**
Conhece e usa sistematicamente	3,9	0,7	2,9	2,5	11,4
Conhece e começa a usar	17,1	3,4	20,4	30,0	29,5
Conhece, mas não usa	53,7	62,2	48,9	47,5	48,9
Não conhece	25,3	33,8	27,7	20,0	10,2

Tabela 8- Conhecimento do Projeto SPICE [MCT02].

Conhecimento do projeto SPICE - Software Process Improvement and Capability dEtermination (Technical Report ISO/IEC TR 15504)					
Categorias	**Total**	**Micro**	**Pequena**	**Média**	**Grande**
Conhece e usa sistematicamente	1,0	2,0	-	-	1,1
Conhece e começa a usar	3,2	1,4	3,0	5,1	5,7
Conhece, mas não usa	56,7	49,0	50,4	59,0	77,3
Não conhece	39,1	47,6	46,6	35,9	15,9

Diante deste fato, podemos afirmar que por falta de utilização das normas ou modelos de qualidade de software, produzimos softwares de qualidade contestável e participando efetivamente da "Crise do Software" [PRESSMAN95].

7.1 Produto de Software

Quando entregamos a um cliente um pacote bem delimitado e identificado, podemos dizer que entregamos um produto [SPINOLA98].

A definição para produto de software segundo a norma IEEE-STD-610 [IEEE90] é:

"O conjunto completo, ou qualquer dos itens individuais do conjunto, de programas de computador, procedimentos, e documentação associada e dados designados para liberação para um cliente ou usuário final" [PAULK95].

7.2 Processo de Software

O conceito de processo de software se baseia no conceito generalizado de processo, que pode ser definido como uma seqüência de estados de um sistema que se transforma [SPINOLA98].

O SEI (*Software Engineering Institute*), da *Carnegie Melon University* propõe o seguinte [SEI]:

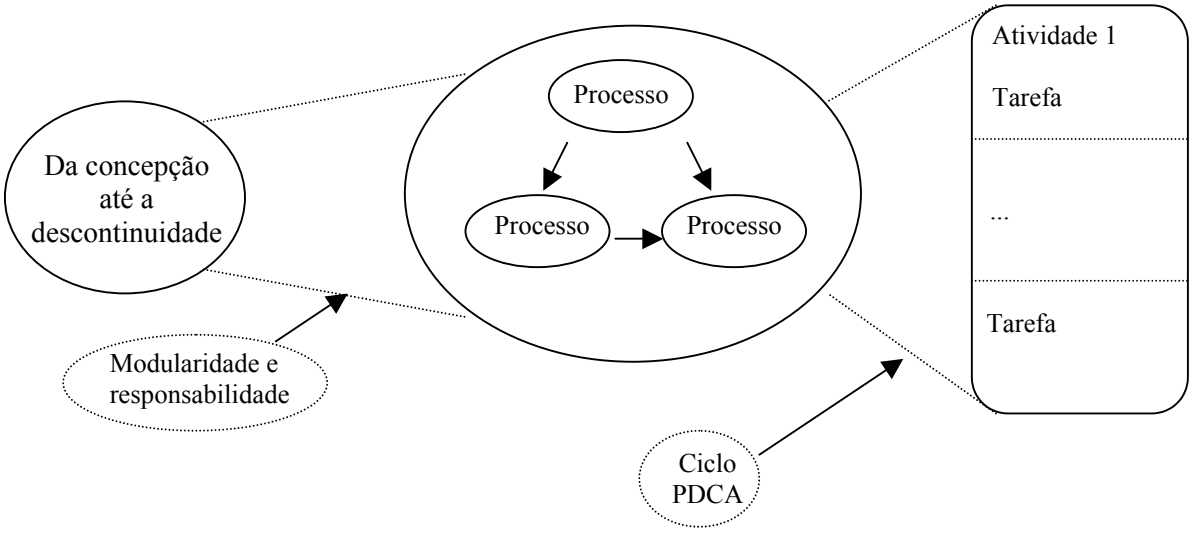
"Um processo é uma seqüência de passos realizados para um dado propósito. Colocado de maneira mais simples, processo é aquilo que você faz. Processo é aquilo que as pessoas fazem, usando procedimentos, métodos, ferramentas, e equipamentos, para transformar matéria prima (entradas) em produto (saída) que tenha valor para o cliente".[PAULK95]

"O processo de software pode ser definido como um conjunto de atividades, métodos, práticas, e transformações que as pessoas empregam para desenvolver e manter software e os produtos associados (ex. planos de projeto, documentos de projeto (design), código, casos de teste, e manual do usuário)".[PAULK95]

8. ISO/IEC 12207

Segundo Machado [MACHADO01], a globalização da economia vem influenciando as empresas produtoras e prestadoras de serviços de software a alcançar o patamar de qualidade e produtividade internacional para enfrentarem a competitividade cada vez maior. A norma internacional NBR ISO/IEC 12207 – Tecnologia da Informação – Processos de Ciclo de Vida de Software [ISO12207: 97] é usada como referência em muitos países, inclusive no Brasil, para alcançar esse diferencial competitivo.

Figura 1- Arquitetura da norma NBR ISO 12207 [MACHADO01].



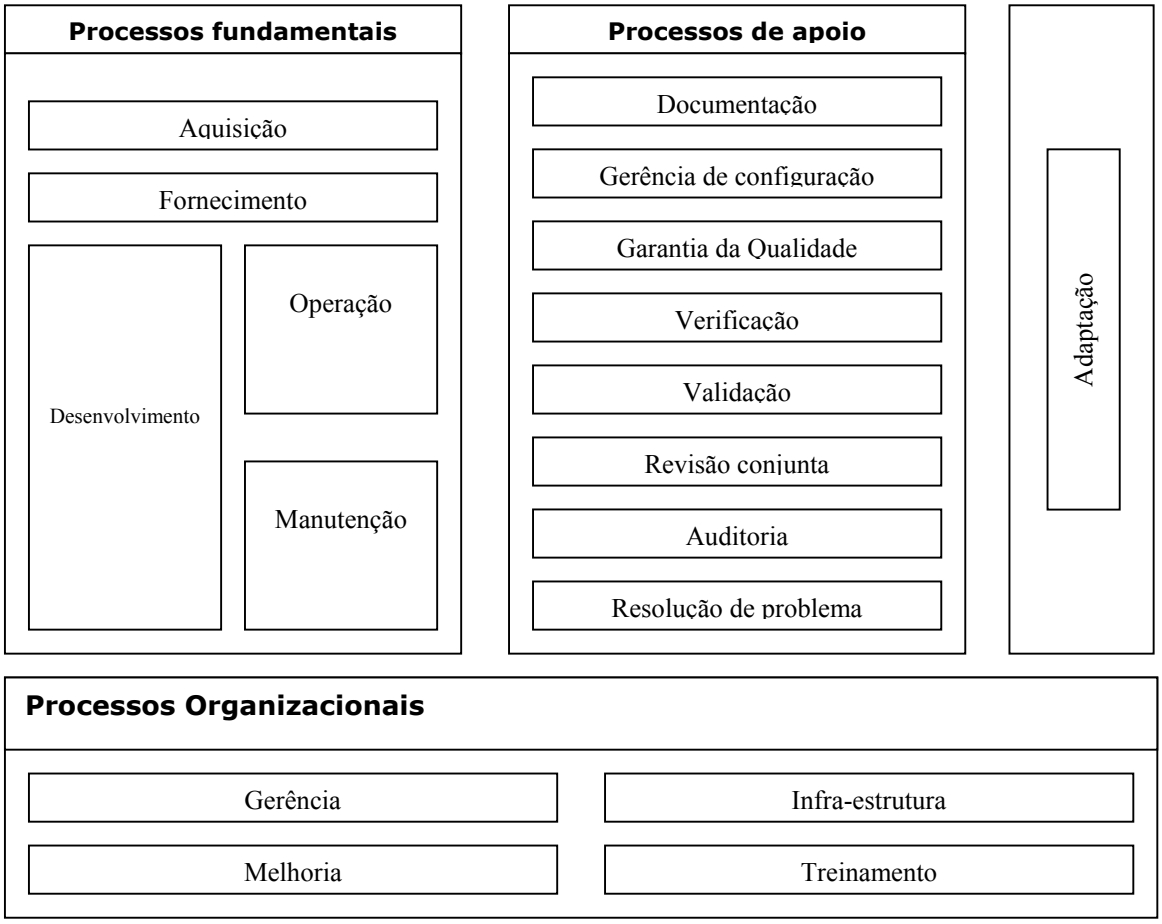
Ela tem por objetivo auxiliar os envolvidos na produção de software a definir seus papéis, por meio de processos bem definidos, e assim proporcionar às organizações que a utilizam um melhor entendimento das atividades a serem executadas nas operações que envolvem, de alguma forma, o software.
A arquitetura descrita na norma (Figura 1) [MACHADO01] utiliza uma terminologia bem definida e é composta de: processos, atividades e tarefas para aquisição, fornecimento, desenvolvimento, operação, e manutenção do software.

A norma estabelece uma arquitetura de alto nível para o ciclo de vida do software que abrange desde a concepção até a descontinuidade do mesmo. Essa arquitetura é baseada em processos-chave e no inter-relacionamento entre eles e segue dois princípios básicos:

- Modularidade: Os processos têm alta coesão e baixo acoplamento, ou seja, todas as partes de um processo são fortemente relacionadas e o número de interfaces entre os processos é mantido ao mínimo.
- Responsabilidade: Cada processo na norma é de responsabilidade de uma "Parte envolvida", que pode ser uma organização ou parte dela. As partes envolvidas podem ser da mesma organização ou de organizações diferentes.

Segundo Machado [MACHADO01], na norma ISO/IEC 12207, os processos que envolvem o ciclo de vida do software são agrupados em três classes que representam sua natureza. Cada processo é definido em termos de suas próprias atividades, e cada atividade é adicionalmente definida em termos de suas tarefas (Figura 2).

Figura 2- Processos de Ciclo de Vida do Software [ISO12207: 97].



8.1 Processos Fundamentais

Os processos fundamentais atendem ao início, à contratação entre o adquirente e o fornecedor e a execução do desenvolvimento, da operação ou manutenção de produtos de software durante o ciclo de vida do software [MACHADO01].

Nas empresas de desenvolvimento, os processos fundamentais acontecem ainda que não estruturados pela adoção de uma norma, mas pela necessidade da realização da prestação de serviço ou do desenvolvimento dentro da própria empresa.

O planejamento é feito de modo precário pela ausência constante de documentação entre o desenvolvedor e o adquirente. Existe resistência em ambas as partes interessadas em gerar documentação, pois normalmente acreditam estar "perdendo tempo" na especificação dos requisitos, no planejamento num todo do projeto. Muitos desenvolvedores partem direto para o desenvolvimento (Codificação) e depois são levados a um processo de correção e manutenção interminável, provocando desgaste da relação comercial estabelecida, com o não cumprimento de prazos de entrega e o custo do projeto que foi orçado por uma estimativa aleatória acumularão prejuízos intangíveis.

8.2 Processos de Apoio

Os processos de apoio auxiliam e contribuem para o sucesso e a qualidade do projeto de software.

Um processo de apoio é empregado e executado quando necessário para documentação, gerência de configuração, garantia da qualidade, processo de verificação, processo de validação, revisão conjunta, auditoria e resolução de problemas [MACHADO01].

A documentação do software será a última tarefa que o desenvolvedor irá se preocupar, sendo tratado como se não tivesse que acontecer antes do desenvolvimento propriamente dito, a fim de ser possível acompanhar se os requisitos do projeto foram atendidos ou se nem foram especificados no momento oportuno. Os processos de verificação e validação ocorrem unilateralmente, ou seja, "este requisito era óbvio, nem precisava mencionar", e para atender as necessidades do adquirente, este processo será repetido por inúmeras vezes, alongando a manutenção e atrasando o funcionamento e atendimento as necessidades do negócio.

8.3 Processos Organizacionais

Os processos organizacionais são empregados por uma organização para estabelecer e implementar uma estrutura constituída pelos processos de ciclo de vida e pelo pessoal envolvido no desenvolvimento do software. Eles são geralmente empregados fora do domínio de projetos e contratos específicos; entretanto, os ensinamentos desses projetos e contratos contribuem para a melhoria da organização, são eles: Processos de Gerência, Infra-estrutura, Melhoria, e Treinamento [MACHADO01].

O processo de gerência depende diretamente do porte da empresa. Em alguns casos existirá a pessoa responsável, em outros o próprio desenvolvedor assumirá o papel, mas não a função, ou seja, aparecerá como responsável, mas devido à carência de tempo e de recursos humanos fará todo o trabalho e não praticará a gerência do projeto.

Para implementar os processos de infra-estrutura, melhoria e treinamento, é fundamental a figura de gerência que exercerá acompanhamento das necessidades do projeto e seus devidos ajustes quanto a estrutura necessária para um desenvolvimento dentro dos requisitos do projeto, do dinamismo necessário para melhoria contínua do processo e os devidos treinamentos para adequação das tecnologias especificadas nos requisitos.

8.4 Processos de Adaptação

O processo de adaptação define as atividades necessárias para adaptar a norma para sua aplicação na organização ou em projetos. A adaptação deve ser executada com base em alguns fatores que diferenciam uma organização ou projeto de outros, dentre os quais a estratégia de aquisição, modelos de ciclo de vida de projeto, características de sistemas e software e cultura organizacional. A existência desse processo permite que a norma seja adaptável a qualquer projeto, organização, modelo de ciclo de vida, cultura e técnica de desenvolvimento [MACHADO01].

A adaptação inicia-se com a disseminação da na norma promovendo o conhecimento de sua estrutura, a necessidade de sua adoção e os principais benefícios e medindo os impactos que ela trará a partir de sua implantação. Diante da realidade e do "caos" em que as empresas de desenvolvimento vivem com a falta de estrutura e planejamento, os empresários desenvolvedores se voltarão para o processo de desenvolvimento de software com qualidade, observando a "fatia" de mercado que poderá expandir seus negócios e ser ponto de referência no mercado de software brasileiro.

Atualmente já existe um movimento das empresas para adoção de normas e modelos de maturidade do processo de desenvolvimento de software, buscando melhor produtividade e com ênfase em promover uma reengenharia nos processos de desenvolvimento de software, que até então eram basicamente vindos da experiência dos "desenvolvedores de código" e não de gestores de projetos de grande expressão, e que assumem papel de alta relevância nas empresas para se obter vantagens competitivas num mercado que busca a informação certa no momento certo.

9. CONCLUSÃO

Todas as normas e modelos de qualidade para software têm por objetivo buscar organização e melhoria continua no processo de desenvolvimento de software. Porém fica identificado que a ISO/IEC 12.207, possui pontos importantes a serem ressaltados:

- Adequação aos princípios e objetivos da Engenharia de Software.
- Possibilita então ser fator crítico para o sucesso no desenvolvimento de software.
- Adaptabilidade na inexistência de alguns processos, dependendo do porte da empresa desenvolvedora de software [ISO/IEC 12207: 97].
- Com a sua adoção é possível contribuir para o fim da "Crise do Software".
- Com os processos de desenvolvimento de software controlados, documentados e gerenciados o desenvolvedor poderá assumir projetos de alta complexidade, aliados a técnica e criatividade, pois terá mais chance de sucesso.
- Melhor capacitado e provedor de metodologias que levam ao desenvolvimento de software com qualidade, o desenvolvedor poderá criar soluções que atendam as necessidades e os requisitos da empresa, contribuindo para criação de vantagens competitivas, sustentando as bases estratégicas das organizações.

10. REFERENCIAS BIBLIOGRÁFICAS

[FIORINI98] FIORINI, SOELI T., et al. *Engenharia de Software com CMM*, Rio de Janeiro, Ed. Brasport, 1998.

[IEEE90] IEEE STD. 610 12-1990, *IEEE Standard Glossary of Software Engineering Terminology*, IEEE, Piscataway, NJ, 1997.

[ISO12207: 97] NBR ISO/IEC 12207:1997, *Tecnologia de Informação – Processos de Ciclo de Vida de Software*, Rio de Janeiro, ABNT – Associação Brasileira de Normas Técnicas.

[LEE02] LEE, RICHARD C. e TEPFENHART, WILLIAM M., *UML e C++ - Guia de desenvolvimento orientado a objeto*, São Paulo, Ed. Makron Books, 2002.

[MACHADO01] MACHADO, CRISTINA ÂNGELA FILIPAK in WEBER, KIVAL CHAVES, et al. *Qualidade e Produtividade em Software*, São Paulo, Ed. Makron Books, 2001.

[MAFFEO92] MAFFEO, BRUNO, *Engenharia de Software e Especificação de Sistemas*, Rio de Janeiro, Ed. Campus, 1992.

[MARTIN91] MARTIN, JAMES*,Engenharia da Informação*, Rio de Janeiro, Ed. Campus, 1991.

[MCT02] MINISTÉRIO DA CIÊNCIA E TECNOLOGIA, Secretaria de Política de Informática, *Qualidade e Produtividade no Setor de Software Brasileiro*, Brasília, N.4, 2002.

[PÁDUA03] FILHO, WILSON DE PÁDUA PAULA, *Engenharia de Software*, Rio de Janeiro, Ed. LTC, 2003.

[PAULK95] PAULK, M.C. et al. *The Capatibility Maturity Model – Guidelines for improving the software process*, Addison Wesley, SEI series, 1995.

[PRESSMAN95] PRESSMAN, ROGER S., *Engenharia de Software*, São Paulo, Ed. Makron Books, 1995.

[PRESSMAN02] PRESSMAN, ROGER S., *Engenharia de Software*, Rio de Janeiro, Ed. McGraw-Hill, 2002.

[REZENDE99] REZENDE, DENIS ALCIDES, *Engenharia de Software e Sistemas de Informações*, Rio de Janeiro, Ed. Brasport, 1999.

[SEI] SEI, *Software Engineering Institute*, *Carnegie Melon University, http://www.sei.cmu.edu.*

[SOMMERVILLE03] SOMMERVILLE, IAN, *Engenharia de Software*, São Paulo, Ed. Pearson Education, 2003.

[SPINOLA98] SPINOLA, MAURO DE MESQUITA, *Diretrizes para o desenvolvimento de software de sistemas embutidos*, Tese de Doutorado, USP - São Paulo, 1998.