A Melhoria de Processo de Software Baseada no CMM: Um Enfoque para Empresas de Pequeno Porte no Brasil

Gizelle S. Lemos •
Angelita M. Segovia ♦
Alfredo Colenci Neto *
Dr. Penido Stahlberg Filho *
Phd Fredy Valente *
Dr. Edson W. Cazarini ♦

* S&V Consultoria e Tecnologia
Caixa Postal 781
CEP 13560-320
São Carlos -SP - Brasil
{neto, penido, fredy}@svconsultoria.com.br
♦ Universidade de São Paulo – São Carlos
Av. Dr. Carlos Botelho
São Carlos -SP - Brasil
angelita@svconsultoria.com.br
casarini@sc.usp.br

• Universidade Federal de São Carlos
Rodovia Washington Luiz, km 243
São Carlos – SP- Brasil
gizelle@dc.ufscar.com.br

Resumo

As mudanças no plano político vêm sendo sistematicamente acompanhadas pelas diretrizes do Governo Brasileiro. Programas de qualidade, incluindo a certificação dos processos e produtos de software, hoje caracterizados como competências para competitividade, tornar-se-ão indispensáveis para a sobrevivência em longo prazo no mercado internacional. Pesquisas constataram uma forte relação entre qualidade do software e seu processo de desenvolvimento.

Na questão da avaliação e melhoria do processo, o CMM (Capability Maturity Model) qualifica a capacitação dos processos de software, com o objetivo de avaliar o estado atual e definir ações para a melhoria evolutiva desses processos. Ele também orienta na identificação das práticas-chave requeridas para aumento da maturidade dos processos. Embora 36% das empresas brasileiras sejam de pequeno porte, nenhuma recebeu a certificação CMM. Sua aplicação, portanto, é justificada à medida que aumenta a possibilidade das mesmas participarem do mercado mundial, onde são exigidas a capacitação e maturidade dos processos de software

1. Introdução

Aos valores econômicos clássicos – capital, trabalho e terra, uma nova dimensão foi adicionada – a do conhecimento. Na chamada "sociedade da informação", o quadro mundial passa por fortes e aceleradas transformações em suas estruturas políticas, econômicas, sociais e culturais, e verdadeiras revoluções nas tecnologias de informação e comunicação.

Considerando que o setor de software será, segundo pesquisas internacionais, responsável pelos maiores índices de crescimento na economia global nos próximos anos, o Brasil está participando do movimento que consiste em uma estratégia para o aumento da competitividade baseada em qualidade, custos e eficiência.

Com taxa média anual de crescimento da receita de 19% sobre os valores correntes, o setor de software

brasileiro apresentou um melhor desempenho na década de 90, quando comparado ao hardware, que cresceu 6% ao ano no mesmo período. Trata-se de um mercado de US$2,5 bilhões provenientes da comercialização de software das empresas desenvolvedoras nacionais, acrescido de US$1,2 bilhões estimado para a importação de software, resultante da remessa em direitos autorais.

Neste contexto, o Brasil está procurando alcançar padrões internacionais efetivos em qualidade e produtividade no setor de software. Grandes investimentos têm ocorrido na área com fomento da Sociedade Brasileira para Promoção da Exportação de Software – SOFTEX, órgão governamental que tem como um dos objetivos transformar o Brasil em um centro de excelência na produção e exportação de software.

2. Problemas de empresas que desenvolvem software

Segundo [18], em muitas organizações, os projetos são entregues muito além do tempo planejado e com o dobro do custo estimado. Os benefícios dos métodos e ferramentas não podem ser alcançados por meio da indisciplina ou por meio de um projeto caótico. Em alguns casos, a organização não possui infra-estrutura e suporte necessários para auxiliarem nos projetos.

Na ausência de um processo organizado de desenvolvimento de software, os resultados dependem da existência de algumas avaliações individuais para projetos próximos. De acordo com [18], a melhoria contínua pode ocorrer somente através de esforços focados e sustentados pela construção de uma infra-estrutura de engenharia de software e práticas de gerenciamento.

3. Qualidade aplicada ao software

Kan em [9], apresenta uma perspectiva histórica a respeito da Engenharia de Software, a qual é apresentada a seguir:

- Era Funcional – anos 60: Aprendeu-se a usar a tecnologia de informação para suprir as necessidades institucionais e começar a integrar o software nas operações diárias das organizações.
- Era do Método – anos 70: Devido aos atrasos nos planos e ultrapassagem nos custos, a maior preocupação nessa fase foi o estabelecimento de planejamento e controle de projetos. Foi quando os modelos de ciclo-de-vida do software foram introduzidos e analisados.
- Era do Custo – anos 80: O custo do hardware começou a cair e a tecnologia de informação tornou-se acessível às pessoas, não mais somente às organizações. A competição das indústrias tomou um rumo diferente, uma vez que as aplicações de baixo custo puderam ser largamente implementadas. Assim, a importância da produtividade no desenvolvimento de software aumentou significativamente. Foi no final dessa década que se reconheceu a importância da qualidade do software.
- Era da Qualidade – anos 90: Com a tecnologia do estado da arte, espera-se atender à demanda dos clientes com a exigência da alta qualidade. Essa exigência é intensificada pela crescente dependência da sociedade pelo software.

A respeito da qualidade de software, há muitos conceitos para defini-la, sob diferentes pontos de vista apresentados na literatura. O Quadro 1 a seguir, apresenta algumas destas definições:

Definição	Referência
Um produto de software apresenta qualidade dependendo do grau de satisfação das necessidades dos clientes sob todos os aspectos do produto.	[16]
É o grau em que o software possui uma combinação desejada de atributos.	[06]
É o grau em que os atributos do software são capazes de desempenhar sua finalidade especificada.	DoD-Std2168
A percepção da qualidade de software é vista principalmente em termos de tempo em que um sistema de software opera corretamente.	[19]
É a conformidade aos requisitos estabelecidos, aos padrões de desenvolvimento documentados e às características implícitas que são aguardadas pelos desenvolvedores de software.	[14]

Quadro 1
Definições sobre Qualidade de Software

Com o objetivo de dar continuidade ao estudo da qualidade do software, o assunto é dividido em duas áreas: qualidade do produto e qualidade do processo de software. Estas áreas são apresentadas a seguir.

3.1. Qualidade do produto de software

De acordo com Endo [02], nos últimos 60 anos, todo esforço esteve concentrado em se obter controle da qualidade do produto final, para que um produto defeituoso não chegasse às mãos do cliente. Trabalhos realizados por McCall, apresentados na norma ISO/IEC 9126 [08], procuram definir um conjunto de características que evidenciam a qualidade do produto final [01]. Essas características são definidas para representar necessidades e desejos daqueles que estão direta ou indiretamente envolvidos com o software, [20]. Porém, estudos recentes demonstraram que a importância dada ao processo de software é uma forma de garantir a construção de um produto de melhor qualidade.

3.2. Qualidade do processo de software

De acordo com Paulk[13], o processo de software é representado por um conjunto seqüencial de atividades, objetivos, transformações e eventos que encapsulam estratégias para o cumprimento da evolução do software.

Para Fiorini [03], conhecer os processos significa conhecer como os produtos e serviços são planejados, produzidos e entregues ao cliente.

Para que um processo de software seja efetivo no cumprimento dos seus objetivos, torna-se necessário um planejamento detalhado, que revele a realidade do ambiente de desenvolvimento do software.

Deve-se considerar aspectos específicos do projeto, como metas e políticas, equipe de desenvolvimento, cronograma, disponibilidade de recursos humanos, técnicos e financeiros.

Além desses aspectos, outros também devem ser relacionados ao processo, como: fluxo de informação, condições específicas para delimitar o início e o fim de cada fase, seqüenciamento das atividades e papéis desempenhados pelas pessoas, [22].

A grande importância dada ao processo de software pode ser vista como forma de garantir a construção de um software de melhor qualidade. Espera-se que um processo controlado de software propicie segurança frente às variações que o produto possa sofrer em relação às suas especificações iniciais, [19].

Para isso, um mecanismo gerencial deve garantir o cumprimento das responsabilidades e a condução do grupo para que se possa desenvolver o sistema de maneira segura e controlada, [09].

4. Melhoria de processo de software

4.1. Capability maturity model (CMM) - SEI/CMU

O SEI *(Software Engineering Institute)* sediado na CMU *(Carnegie Mellon University)*, em Pittsburg, Pennsylvania – Usa, é um centro de pesquisa que foi criado em 1984 pelo Departamento de Defesa dos Estados Unidos (DoD – *Department of Defense*) e é patrocinado pelo OUSD (A&T) (*Office of the Under Secretary of Defense for Acquisition and Technology*). As áreas de atuação do SEI são: capacitação de gerência de software e tecnologia para a engenharia. O SEI focaliza também a transição tecnológica, ou seja, o desenvolvimento e adoção das melhores práticas de engenharia de software.

4.1.1 Estrutura do CMM

De acordo com Humprey [05], o CMM promove nas organizações desenvolvedoras de software, um guia de como alcançar o controle em seus processos de desenvolvimento e manutenção de software, e um modelo de como envolver uma cultura de engenharia de software e práticas de gerenciamento. O CMM foi desenvolvido para orientar organizações na seleção de estratégias de melhoria de processos pela maturidade atual e identificar áreas mais críticas de qualidade de software. A Figura 1 a seguir ilustra os Níveis de Maturidade do CMM:

Figura 1
Níveis de Maturidade do CMM [13]

Como apresentado pela Figura 1, o CMM classifica as organizações em cinco níveis evolutivos de maturidade, cada nível com suas características próprias. Estas características estão apresentadas no Quadro 2 a seguir:

Quadro 2
Visão Geral dos Níveis de Maturidade do CMM [13]

Nível CMM	Características
(1) Inicial	O processo de desenvolvimento é desorganizado e ate caótico. Poucos processos são definidos e padronizados e o sucesso depende de esforços individuais e heróicos dos desenvolvedores.
(2) Repetitivo	Os processos básicos de gerenciamento de projeto estão estabelecidos e permitem acompanhar custo, cronograma e funcionalidade, é possível repetir o sucesso de um processo já utilizado anteriormente.
(3)	Tanto as atividades de

Definido	gerenciamento quanto de engenharia do processo de desenvolvimento de software estão documentadas, padronizadas, e integradas em um padrão de desenvolvimento da organização. Todos os projetos utilizam uma versão aprovada e adaptada do processo padrão de desenvolvimento de software da organização.
(4) Gerenciado	São coletadas medidas detalhadas da qualidade do produto e processo de desenvolvimento de software. Tanto o produto quanto o processo de desenvolvimento são entendidos e controlados quantitativamente.
(5) Otimizado	O melhoramento contínuo do processo é conseguido através de um *feedback* quantitativo dos processos e pelo uso pioneiro de idéias e tecnologias inovadoras.

4.1.2 As áreas chave de processos (KPA's)

Todos os níveis do CMM, com exceção do nível 1, são compostos de um certo número de áreas-chave de processo (*Key Process Areas* ou KPA's), que descrevem os objetivos a serem atingidos, assim como as questões a serem endereçadas para se alcançarem estes objetivos e atingir aquele nível. Todas as áreas chaves são apresentadas a seguir:

Quadro 3
Visão Geral dos Níveis de Maturidade do CMM e suas respectivas KPA's [13]

Nível CMM	Foco	Áreas Chave de Processo
(1)	Esforço Individual	---
(2)	Processo de Gerenciamento de Projetos	Gerenciamento de Requisito Planejamento do Projeto Supervisão e Acompanhamento do Projeto Gerenciamento de Subcontratados Garantia da Qualidade do Software Gerenciamento de Configuração do Software
(3)	Processos de Engenharia e Apoio	Foco do Processo Organizacional Definição do Processo Organizacional Programa de Treinamento Gerenciamento de Software Integrado Engenharia de Produto de Software Coordenação Intergrupos Revisão Conjunta
(4)	Qualidade do Produto e do Processo	Gerenciamento Quantitativo dos Processos Gerenciamento da Qualidade do Software
(5)	Melhoramento Contínuo do Processo	Prevenção de Defeitos Gerenciamento de Mudanças Tecnológicas Gerenciamento de Mudanças no Processo

4.1.3 O caminho para o Nível 2 de maturidade do CMM

A adoção do CMM baseia-se em um processo gradual de aumento da maturidade do desenvolvimento do software. Essa maturidade pode ser traduzida como a probabilidade de entregar sistemas de software dentro dos prazos, utilizando os mesmos recursos planejados e atendendo aos requisitos e qualidade desejados [21].

O objetivo de alcançar o Nível 2 é institucionalizar um processo efetivo de gerenciamento de projetos de software, que permite às organizações repetir suas práticas através de processos implementados para projetos diferentes. Um processo efetivo pode ser caracterizado como aquele que é praticado, documentado, treinado, medido e ser capaz de melhorias.

As organizações com projetos em Nível 2 têm instalado controles de gerenciamento básico de software. Compromissos realísticos de projetos são baseados nos resultados observados em projetos prévios e nas necessidades dos atuais. Um gerente de software rasteia os custos, tempo, funcionalidade e os problemas são identificados. Os requisitos são desenvolvidos e sua integridade é controlada. Projetos padrões são definidos e a organização assegura que os mesmos serão fielmente seguidos.

É apresentado a seguir, um detalhamento das KPA's referentes ao Nível 2 do CMM.

KPA 1 - Gerenciamento de Requisitos: a proposta do Gerenciamento de Requisitos é estabelecer um entendimento comum entre o cliente e a equipe do projeto sobre os requisitos alocados ao software. Envolve o estabelecimento e manutenção dos requisitos de acordo

com as necessidades do cliente. O cliente pode ser interpretado como o grupo de engenharia, marketing, outro interno da organização ou um cliente externo. Este acordo forma a base para estimar, planejar, executar e localizar as atividades do projeto de software ao longo do seu ciclo-de-vida, [13].

KPA 2 - Planejamento do Projeto do Software: o propósito é estabelecer planos para o desempenho e manutenção do projeto de software. Envolve o desenvolvimento de estimativas para o trabalho, estabelecimento de compromissos e de um plano para guiar o projeto. O planejamento começa com a declaração do trabalho e outras metas que definem o projeto (estabelecidos pelas práticas do Gerenciamento de Requisitos). O processo de planejamento inclui passos para estimar o tamanho do produto de trabalho e recursos necessários, identificação e avaliação de riscos. A interação através desses passos deve ser necessária para estabelecer o plano do projeto. Este plano promove a base para o desenvolvimento e gerenciamento das atividades do projeto e encaminha as necessidades do cliente de acordo com os recursos e capacidades do projeto.

KPA 3 - Acompanhamento e Supervisão de Projeto de Software: sua finalidade é promover uma visão adequada do progresso real do projeto, de modo que o gerenciamento possa tomar medidas efetivas quando o desempenho desvia-se do plano proposto. Envolve acompanhar e revisar os resultados e as realizações do software confrontando com as estimativas documentadas, compromissos e planos. Envolve também o ajuste de planos com base nos resultados alcançados. Os mecanismos utilizados podem ser revisões internas com a participação dos desenvolvedores e gerentes e revisões formais com os clientes. Quando ocorrer um desvio entre os planos e os efetivos resultados, deve-se alterar a forma como o trabalho esta sendo desempenhado ou ajustar os planos.

KPA 4 - Gerenciamento de Subcontratos de Software: sua finalidade é selecionar fornecedores qualificados e gerenciá-los eficazmente. Esse processo envolve estabelecer compromissos, acompanhar e revisar o desempenho e os resultados obtidos. Na seleção e gerenciamento do fornecedor, são necessários documentos como: cláusula do contrato, requisitos do projeto, produtos a serem entregues, padrões e procedimentos a serem seguidos.

KPA 5 - Garantia de Qualidade de Software: sua proposta é promover o gerenciamento, com visibilidade do processo que está sendo utilizado e dos produtos que estão sendo construídos. Envolve revisões e auditorias nos produtos de software e nas atividades para assegurar que estão em conformidade com os padrões e procedimentos aplicados.

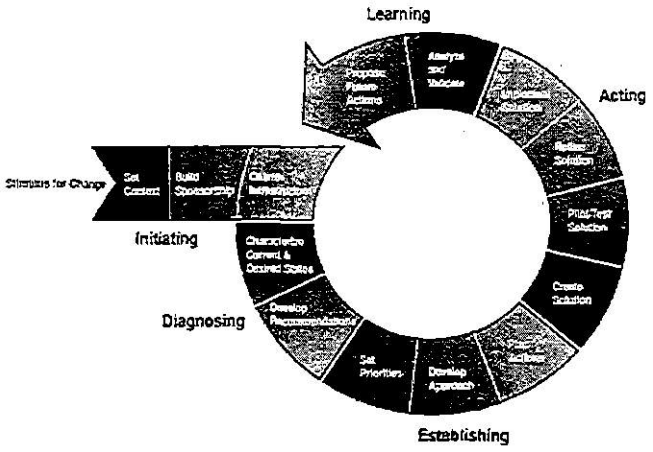
KPA 6 - Gerenciamento da Configuração de Software: sua proposta é estabelecer e manter a integridade dos produtos do projeto ao longo do ciclo-de-vida. Envolve identificar os itens de configuração, controlar sistematicamente as alterações e manter a integridade da configuração ao longo do ciclo-de-vida. Utiliza linhas de referência (*baselines*) que servem como um marco no ciclo-de-vida do software. Os itens que passam por uma *baseline* podem ser alterados somente através de procedimentos formais de controle de mudanças.

O SEI estabeleceu uma proposta que descreve fases, atividades e recursos necessários para tomar iniciativas de melhoria de processo com sucesso, [12]. De acordo com Fiorini [03], esta proposta é similar ao ciclo de melhoria PDCA (*Plan, Do, Check and Act* – Planejar, Fazer, Verificar e Agir) e é denominada IDEAL.

4.2. O modelo IDEAL

O modelo IDEAL (*Initiating, Diagnosing, Estabilishing, Acting and Learning* – Inicialização, Diagnóstico, Estabelecimento, Ação e Lições Aprendidas) versão 1.1, descreve uma abordagem de gerência que ajuda as organizações desenvolvedoras de software a melhorarem seu processo de desenvolvimento. Este modelo estabelece um programa de melhoria contínua de processo de software. A Figura 2 a seguir, apresenta sua estrutura:

Figura 2
Estrutura do Modelo IDEAL[13]



4.2.1 Estrutura do modelo IDEAL

De acordo com Endo [02], cada uma das fases é realizada através da execução de uma série de atividades descritas a seguir:

Fase 1-Inicialização: deve-se articular a aplicação dos esforços para satisfazer às necessidades de negócio, assegurar o suporte ao gerenciamento e colocar em prática

uma infra-estrutura para gerenciar os detalhes de implementação do processo de melhoria.

- Estímulo para Mudanças: envolve definir as necessidades de negócio que alavancam as mudanças nas práticas da organização. Quando as necessidades de mudanças são evidenciadas por razões de negócio, é mais fácil convencer a organização como um todo, e existem maiores chances de sucesso.
- Estabelecimento do Contexto: após estarem definidas as razões para melhoria, a organização estabelece o contexto para o trabalho que vai ser realizado. Isso significa definir onde os esforços se enquadrarão na estratégia de negócio organizacional, que metas e objetivos serão afetados pelas mudanças, como isso vai incidir sobre trabalhos futuros e quais os resultados esperados.
- Definição de Patrocinador: o patrocinador deve ajudar a manter o compromisso nos momentos de dificuldades e garantir os recursos essenciais utilizados durante a melhoria.

Fase 2-Diagnóstico: deve-se desenvolver um entendimento do trabalho para a melhoria.

- Fornecimento de Infra-Estrutura: envolve definir um mecanismo para gerenciar os esforços para detalhes de implementação, e também fornecer um contrato que documente e esclareça as expectativas, e descreva as atividades e responsabilidades.
- Caracterização dos Estados Atual e Desejado: uma vez identificado o estado atual, pode-se usar o CMM para definir o desejado.
- Desenvolvimento de Recomendações: as atividades da fase de diagnóstico são desempenhadas pelas pessoas mais experientes ou por especialistas na organização. Suas recomendações são baseadas nas decisões tomadas pelos gerentes-chefe e patrocinadores.

Fase 3-Estabelecimento: deve-se desenvolver um plano detalhado do trabalho, como por exemplo: prioridades e restrições do ambiente operacional. A seguir, uma abordagem que honre esses fatores, ações específicas, prazos, produtos liberáveis e responsabilidades são incorporadas ao plano de ação.

- Estabelecimento de Prioridades: envolve limitação de recursos, dependências entre as atividades recomendadas, intervenção de fatores externos e prioridades globais da organização.
- Desenvolvimento de Abordagem: Envolve a compreensão do escopo do trabalho com o conjunto de prioridades, levando a desenvolver uma estratégia de acompanhamento do trabalho e identificação da disponibilidade de recursos.
- Planejamento das Ações: deve-se desenvolver um plano detalhado de implementação, o qual inclui cronograma, tarefas, prazos, recursos, responsabilidades, medidas, riscos e estratégias.
- Fase 4-Ação: envolve implementar o trabalho que foi contextualizado nas fases anteriores.
- Criação de Solução: desenvolver a melhor solução destinada às necessidades previamente identificadas na organização.
- Teste da Solução: a solução deve ser testada, pois por mais que ela seja bem elaborada, raramente funciona como planejado.
- Refinamento da Solução: a solução pode ser modificada para refletir o conhecimento e a experiência obtidos.
- Implementação da solução: deve-se implementar a solução por toda organização, utilizando algumas abordagens, como por exemplo: just-in-time.

Fase 5-Lições: nessa fase, toda experiência adquirida é revista para determinar o que foi cumprido e como a organização pode implementar melhorias mais eficientemente no futuro.

- Análise e Validação: essa atividade responde a uma série de perguntas: de que maneira o esforço cumpriu a proposta esperada?, o que funcionou melhor?, o que pode ser feito para melhorar a eficiência?. Os dados são coletados, analisados, resumidos e documentados. As necessidades identificadas na fase de inicialização são examinadas para verificar se foram ou não satisfeitas.
- Proposta de Futuras Ações: as recomendações baseadas na análise e validação são desenvolvidas e documentadas.

5. Empresas brasileiras desenvolvedoras de software

Questões relacionadas ao planejamento estratégico, programas e sistemas da qualidade (de produtos e de processos de software), incluindo sua certificação, hoje caracterizadas como competências qualificadoras para competição no mercado global, tornar-se-ão um conjunto de competências básicas para sua sobrevivência a longo prazo no mercado internacional.

De acordo com QUEIROZ (2000), do Ministério da Ciência e Tecnologia, embora as empresas nacionais de software estejam aumentando seus quadros com mão-de-obra especializada e busquem corrigir falhas ouvindo os clientes, a procura por programas de melhoria de qualidade ainda é baixa. Porém, pode-se observar, de acordo com o Gráfico 1, um número crescente de empresas em processo de implantação a cada ano, sendo que, mais da metade desses programas foram implantados a partir de 1997.

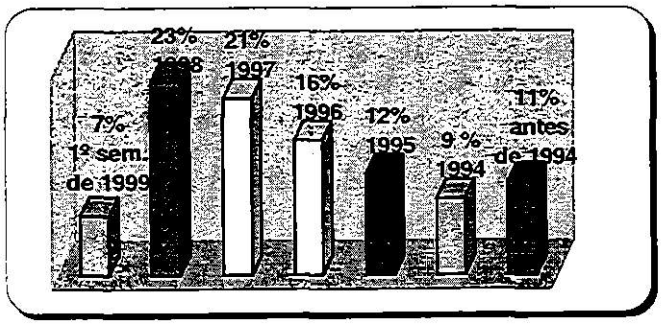


Gráfico 1

Distribuição das empresas, segundo ano de implantação de programa de Qualidade Total, Sistema de Qualidade ou similar [10]

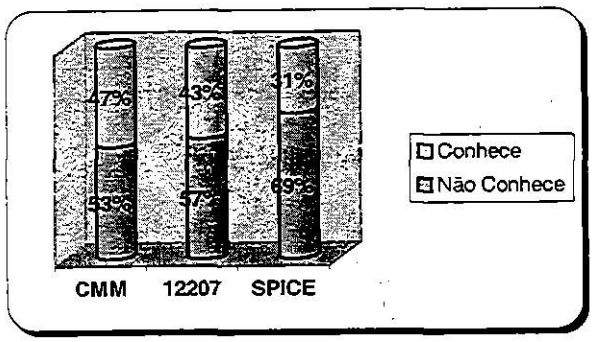


Gráfico 2

Conhecimento de Modelos e Normas da Qualidade de Processos [10]

O software produzido no Brasil está se aproximando do padrão internacional, o que já permite às empresas concorrerem mundialmente. Isso ocorre mesmo sem o conhecimento pleno por boa parte das empresas de normas e projetos relacionados ao tema, como:

- SPICE, que é um projeto que foi estabelecido em junho de 1993 pela ISO/IEC JTC1/SC7 (Subcomitê de Engenharia de Software) com três objetivos principais: auxiliar o desenvolvimento de uma Norma Internacional para avaliação de processos de software; coordenar e analisar utilizações desta futura Norma para subsidiar revisões antes de sua publicação; e disseminá-la no mercado;
- Norma ISO/IEC 12207, Information Technology – Software Life Cycle Process, que define os processos de software, apresentando framework para processo de ciclo-de-vida com terminologia bem definida,
- CMM (Capability Maturity Model)

Além do CMM ser um modelo para avaliação da maturidade dos processos de software de uma organização, ele também orienta seu usuário à identificação das práticas-chave, que são requeridas para aumentar a maturidade desses processos, de acordo com o gráfico 2 Tabela 4, tal conceito é conhecido por 47% das empresas brasileiras, sendo usado por 21% dessas.

Quadro 4

Conhecimento dos Modelos CMM e SPICE [10]

Categoria	CMM		SPICE	
	N.º	%	N.º	%
Conhece e usa sistematicamente	8	1.8	1	0.2
Conhece e começa a usar	36	8.1	14	3.2
Conhece, mas não usa	165	37.2	301	27.3
Não conhece	234	52.8	308	69.4
Base	**443**	**100**	**404**	**100**

A aplicação do CMM nos processos de desenvolvimento de software em empresas brasileiras de pequeno porte é justificada à medida em que aumenta significativamente a possibilidade de participarem do mercado mundial, junto ao qual é exigido a certificação em sistemas de qualidade.

6. Estudo de caso: uma estratégia para a implantação do CMM Nível 2 apoiado pelo modelo IDEAL

6.1. O ambiente

O estudo de caso foi conduzido em uma empresa brasileira de pequeno porte desenvolvedora de soluções de software. A empresa em questão, conta atualmente com cerca de vinte desenvolvedores e três gerentes de projetos. A implementação do CMM Nível 2 foi encarada por seus diretores como um novo projeto, desta forma, equipes foram formadas e cronogramas definidos.

O atingimento deste nível costuma ser o mais difícil e o que despende mais tempo e esforços, uma vez que a organização evolui do nível caótico de desenvolvimento, ou seja, sem padrões ou metodologias, para seguir um modelo padronizado de garantia da qualidade de seus processos de software.

A seguir são apresentadas as fases de implementação do CMM Nível 2 utilizando-se o Modelo IDEAL como guia para melhoria dos processos de software.

6.2. Fase de inicialização

As melhorias do processo de software devem ser conduzidas por equipes que tenham conhecimentos sólidos em engenharia de software e sistemas de qualidade. Nas pequenas empresas, as equipes poderão ser compostas por membros da própria organização, onde uma pessoa poderá desempenhar vários papéis.

Na empresa em questão, a primeira equipe formada foi o SEPG *(Software Engineering Process Group)*, constituída por um funcionário e dois contratados. Essa equipe foi responsável por conduzir as atividades de manutenção e melhoria do processo de software.

A seguir, foi definido o Comitê Estratégico, responsável em prover os recursos necessários para o projeto, assim como aprovar novos processos e políticas.

O terceiro passo foi a definição das pessoas-chave, responsáveis por fornecer aspectos de gerenciamento e planejamento dos processos atuais.

Após formados os grupos, foram ministradas palestras sobre os conceitos e práticas do CMM, com o objetivo de difundir a idéia de qualidade por toda empresa, diminuindo assim qualquer resistência por parte das pessoas envolvidas no processo.

6.3. Fase de diagnóstico

O objetivo desta fase é conhecer a situação real da empresa em matéria de produção de software, [04].

Por se tratar de uma empresa de pequeno porte, a estratégia utilizada pelo SEPG para levantamento dos processos atuais foi entrevistas individuais. As informações coletadas foram inicialmente gravadas e posteriormente documenta.

Um outro instrumento usado para o levantamento e avaliação da situação da empresa foi a aplicação do Questionário de Maturidade, desenvolvido pelo SEI, que após estudado pelo SEPG, foi aplicado às pessoas relacionadas ao processo de desenvolvimento.

Através das informações levantadas, o SEPG pôde ter uma visão abrangente da atual situação do processo de software da empresa identificando os pontos fortes e fracos relacionados a cada área-chave, propondo então, as melhorias para os pontos fracos.

6.4. Fase de estabelecimento

Nesta fase deve-se desenvolver um plano de ação para o atingimento das áreas-chave de processo requeridas para o Nível 2.

Devido ao fato do CMM não estabelecer como os procedimentos de melhoria devem ser implementados, fica a critério da empresa desenvolver mecanismos para a condução de seus processos de melhoria.

A seguir é apresentada uma visão geral dos padrões para o atingimento das áreas-chave 1 e 2 desenvolvidos e estabelecidos na empresa em questão:

Quadro 5
Padrão de documento 1
Gerência de Requisitos

1.	CLIENTE
2.	PROJETO
3.	EQUIPE
4.	VERSÃO DO DOCUMENTO
5.	CAMINHO (onde o arquivo está salvo)
6.	ENTENDIMENTO DO ESCOPO
7.	LEVANTAMENTO DE REQUISITOS
8.	ESTUDO DE VIABILIDADE
9.	SOLUÇÃO PROPOSTA
10.	CRITÉRIOS DE ACEITAÇÃO

Quadro 6
Padrão de documento 2
Planejamento de Projeto de Software / Declaração de Trabalho

1.	ESCOPO DO TRABALHO
2.	METAS TÉCNICAS
3.	OBJETIVOS TÉCNICOS
4.	DESCRIÇÃO DOS ARTEFATOS A SEREM ENTREGUES
5.	IDENTIFICAÇÃO DOS CLIENTES OU USUÁRIOS FINAIS
6.	PADRÕES A SEREM OBEDECIDOS
7.	RESPONSABILIDADES ATRIBUÍDAS
8.	RESTRIÇÕES E METAS PARA CUSTO E CRONOGRAMA
9.	RESTRIÇÕES E METAS PARA OS RECURSOS

Quadro 7
Padrão de documento 3
Planejamento de Projeto de Software / Plano de Desenvolvimento

1.	ESCOPO DO PROJETO
2.	CICLO DE VIDA
3.	ARTEFATOS
4.	ESTIMATIVAS
5.	CRONOGRAMA
6.	RISCOS
7.	FACILIDADES E FERRAMENTAS (Infra-estrutura necessária para o desenvolvimento do projeto)

6.5. Fase de implementação

Fase em que são colocados em prática os planos de ação definidos anteriormente. Isso pode ser feito através da supervisão e acompanhamento de projetos-piloto na organização.

Para o estudo de caso, os padrões de documentos acima foram utilizados em cinco projetos de pequeno porte.

6.6. Fase de lições aprendidas

Nesta fase, todas as práticas bem sucedidas são documentadas, servindo como base para projetos futuros.

Como, na empresa estudada, a implementação do CMM Nível 2 é recente, ainda não se fez uso dos projetos documentados como base para estimativas em novos projetos.

7. Conclusões

Neste artigo, foi proposta uma estratégia para implementar um processo de desenvolvimento e manutenção do software. Esta proposta seguiu algumas diretrizes que um processo deve ter, indicando como avaliar se as metas desejadas estão sendo satisfeitas ou não.

No estudo de caso proposto, sugeriu-se primeiramente entender o funcionamento dos processos de desenvolvimento da empresa, verificando-se quais precisaram ser modificados, adicionados e principalmente, documentados de acordo com as áreas-chave.

Finalmente, o processo de implementação relatado, poderá servir como base para a criação de um modelo de referência/metodologia para empresas de pequeno porte que vivam a mesma dificuldade.

8. Bibliografia

[01] CAVANO, J. P.; MCCALL, J. A. (1978). *A Framework for the Measurement of Software Quality*, Proceeding ACM Software Quality Assurence Workshop, Novembro.

[02] ENDO, Cristina (1998). *Uma Estratégia para Iniciar Melhoria de Processo de Software. Dissertação de Mestrado*, 113 páginas, Escola de Engenharia de São Calos, Universidade de São Paulo.

[03] FIORINI. S. T. (1998).*Engenharia de Software com CMM*, Brasport, Rio de Janeiro.

[04] GRADY. R. B. (1997). *Successful Software Process Improvement*, Prentice-Hall, Nova Jersey.

[05] HUMPYREY. W.S.; SWEET. W. L. (1987a). *Charactering the software process: a maturity framework, Software Engineering Institute*, CMM/SEI-87-TR-11, June.

[06] IEEE (1983). *IEEE Standard Glossary of Software Engineering Terminology*, New York: IEEE, ANSI/IEEE Std 729.

[07] IEEE (1991). *IEEE Standard Glossary of Terminology in Software Engineering*. In: IEEE Software Engineering Standard Collection, p. 07-83.

[08] ISO/IEC 9126 (1994). *Tecnologia de Informação-Avaliação do Produto de Software - Características de Qualidade de Diretrizes para o seu Uso* - Versão brasileira em processo de votação pela ABNT com numeração: NBR-ISO/IEC 9126, junho.

[09] KAN, S. H. (1995). *Metrics and Models in Software Quality Engineering*, Addison-Wesley, EUA.

[10] MCT (1999). *Qualidade e Produtividade no Setor de Software Brasileiro*. Ministério da Ciência e Tecnologia. Número 3. 184 pg. Brasil, Brasília.

[11] PALADINI, E. P. (1990). *Controle de Qualidade - Uma Abordagem Abrangente*, Editora Atlas.

[12] PAULK, M. C. (1994). *A Comparison of ISO 9001 and the Capability Maturity Model for Software*, Software Engineering Institute, Carnegie Mellon University, CMU/SEI-94-TR-12, July.

[13] PAULK, M. C.; GARCIA, S. M.; CHRISSIS. M. B., WEBER, C. V.; BUSH, M. (1993). *Key practices of the Capability Maturity Model*, Version 1.1, Software Engineering Institute, Carnegie Mellon University, CMU/SEI-93-TR-25. ESC-TR-93-178, February.

[14] PRESSMAN, R. S. (1994). *Software Engineering - A Practtioner's Approach*, Ed. Européia, McGraw-Hill.

[15] QUEIROZ, Luiz (2000). *Software Nacional não tem ISO 9000.* ComputerWorld. http://www.uol.com.br/computerworld/technology/0007/0007isso.htm, pesquisa realizada em 17/07/2000.

[16] SANDERS, J.; CURRAN, E. (1994). Software Quality - *"A Framework for Success in Software Development and Suport"*, Addison-Wesley.

[17] SEI (1997). *Process Maturity Profile of the Software Community* 1996-year end Update, Software Engineering Institute.

[18] SIEGEL, J. A. L. (1990). *National Software Capacity: Near – Term Study*, Software Engineering Institute, CMU/SEI-90TR-12, ADA226694, May.

[19] SMITH, D. J. WOOD, K. B. (1989). *Engineering Quality Software - A Review of Current Practices Standards and Guidelines including new Methods and Developement Tool*, Elsevier Science Publishers Ltda.

[20] TSUKUMO, N. A. (1995). *Modelos de Processo de Software - Visão Global e Análise Comparativa*, Fundação CTI - Brasil, Campinas.

[21] MARCINIATI, J.J. (1994). Encyclopedia of Software Engineering, Wiley Intersciense Publications, vII, p.851-869.

[22] PIRES,A.; MENDES, J.R.B. (1999). CMM: O Difícil é dar o Primeiro Passo para a Qualidade, Developers Magazine, Fevereiro.