

**UNIVERSIDADE NOVE DE JULHO – UNINOVE  
CURSO DE ESPECIALIZAÇÃO LATO SENSU EM  
ENGENHARIA DE SOFTWARE**

**DANILO DE SOUSA ABREU**

**UMA PROPOSTA PARA GARANTIA DA QUALIDADE  
DE SOFTWARE ALINHADA COM O CMMI**

**SÃO PAULO  
2015**

**DANILO DE SOUSA ABREU**  
**RA 614106799**

**UMA PROPOSTA PARA GARANTIA DA QUALIDADE  
DE SOFTWARE ALINHADA COM O CMMI**

Monografia apresentada à Universidade Nove de Julho – UNINOVE, como requisito parcial para obtenção do grau de Especialista em Engenharia de Software.

Professor Marcos Ribeiro, MS – Orientador.

**SÃO PAULO**  
**2015**

**DANILO DE SOUSA ABREU**  
**RA 614106799**

**UMA PROPOSTA PARA GARANTIA DA QUALIDADE  
DE SOFTWARE ALINHADA COM O CMMI**

Monografia apresentada à Universidade Nove de Julho – UNINOVE, como requisito parcial para obtenção do grau de Especialista em Engenharia de Software, pela Banca examinadora, formada por:

São Paulo, \_\_\_\_\_ de Abril de 2015.

---

Presidente: Professor Marcos Antonio Ribeiro, MS – Orientador UNINOVE.

---

Membro: Professora Olivia Ramos Moraes Braga – UNINOVE.

Dedico este trabalho aos meus pais, Dona  
Margarida Barroso de Sousa e senhor Serafim  
Pereira de Abreu.

## RESUMO

---

Dentro do cenário atual onde a competitividade no setor de desenvolvimento de software está cada vez mais acirrada, obter um processo de qualidade que promove uma cultura organizacional de garantia e controle da qualidade dos seus produtos torna-se essencial não apenas para a sobrevivência da empresa, mas para ratificar uma presença mais forte no mercado. Como analisado neste trabalho, temos como premissa que a qualidade de software é decorrente da qualidade dos processos de sua construção, logo, estabelecer processos com qualidade seguindo modelos amadurecidos e reconhecidos pelo mercado deve ser, portanto, o foco das organizações, que buscam qualidade em seus produtos e serviços. Este trabalho apresenta uma visão geral sobre o processo de qualidade de software, com uma fundamentação teórica, revisando os conceitos de qualidade, qualidade de software e qualidade de processo, assim como, uma descrição geral sobre alguns dos modelos mais conceituado que descrevem modelos de processos e conjunto de características pertinentes à qualidade em produtos e serviços de Software. É apresentado com ênfase o modelo CMMI-DEV, considerado um dos melhores modelos de qualidade para empresas de produtos e serviços. Com base na área de processo PPQA do CMMI-DEV, é apresentada uma proposta de garantia da qualidade, como um conjunto de atividades, de modo a propor um guia geral para as organizações que buscam conformidade com o CMMI-DEV e um modelo de maturidade para garantir qualidade em seus produtos e, ou serviços de software.

**Palavra chaves:** Garantia da qualidade, qualidade de software, CMMI-DEV, ISO/IEC 25010.

## ABSTRACT

---

*In the current scenario where competitiveness in the software development industry is increasingly fierce, get a quality process that promotes an organizational culture assurance and quality control of their products becomes essential not only for the survival of the company but to confirm a stronger share market presence. As discussed in this work, we have the assumption that software quality is due to the quality of the processes of its construction, so establish processes with quality following matured and recognized by the market models should therefore be the focus of organizations that seek quality on their products and services. This work shows an overview of the software quality process, with a theoretical basis, reviewing the concepts of quality, software quality and process quality as well as an overview of some of the most renowned models that describe process models and set of relevant characteristics of quality in products and software services. It is presented with emphasis CMMI-DEV model, considered one of the best quality models for products and services companies. Based on the process area PPQA from CMMI-DEV, a quality assurance proposal is presented as a set of activities in order to propose a general guide for organizations seeking compliance with the CMMI-DEV and a maturity model to ensure quality in its software products and, or services.*

**Keywords:** *Quality assurance, Software quality, CMMI-DEV, ISO/IEC 25010.*

## LISTA DE FIGURAS

---

Figura 1. Modelo McCall (1977) .....	14
Figura 2. Adaptado de ISO/IEC 25010 (2011) .....	16
Figura 3. Níveis de maturidade do CMMI. Adaptado de CMMI (2010) .....	25
Figura 4. Ilustração, um exemplo de área de processo. CMMI (2010). .....	25
Figura 5. Estrutura de atividades do método proposto .....	35

## **LISTA DE TABELAS**

---

<b>Tabela 1. Áreas de processo agrupadas por nível. Adaptado de CMMI (2010)..</b>	<b>27</b>
<b>Tabela 2. Papéis previstos. Adaptado de MANUVANNAN (2010).....</b>	<b>31</b>
<b>Tabela 3. Análise de aderência da proposta .....</b>	<b>36</b>



## SUMÁRIO

---

<b>1. INTRODUÇÃO .....</b>	<b>10</b>
<b>2. QUALIDADE .....</b>	<b>13</b>
2.1. Conceito .....	13
2.2. Qualidade de Software .....	13
2.3. Qualidade de processo .....	17
2.4. Os pilares da qualidade .....	18
2.4.1. Planejamento da qualidade .....	18
2.4.2. Garantia da qualidade .....	19
2.4.3. Controle da qualidade .....	19
<b>3. MODELO DE MATURIDADE CMMI (<i>Capability Maturity Model Integration</i>).....</b>	<b>20</b>
3.1. CMM .....	20
3.2. CMMI .....	22
3.3. CMMI-DEV ( <i>for development</i> ) .....	23
3.3.1. Níveis de maturidades contidas no CMMI-DEV .....	24
3.3.2. Garantia da qualidade do produto e processo (PPQA).....	28
<b>4. VISÃO GERAL PARA ESTABELECE O PPQA .....</b>	<b>29</b>
4.1. Artefatos .....	30
4.2. Atividades .....	30
<b>5. CONCLUSÃO.....</b>	<b>37</b>
5.1. Dificuldades encontradas .....	37
5.2. Considerações finais .....	37
5.3. Trabalhos futuros .....	38

## 1. INTRODUÇÃO

Dentro de um cenário de grande competitividade, e cada vez mais acirrada, as organizações buscam diferenciais para os seus produtos ou serviços. Cada vez mais o conjunto de relação custo benefício, versus curto prazo e produtos ou serviços com qualidade são almejados pelos consumidores. Para as organizações que pretendem manterem-se vivas dentro deste cenário, devem estar cada vez mais ligadas as novas necessidades buscadas por esse mercado exigente e extremamente fértil atualmente.

No mercado de desenvolvimento de soluções de software obter diferencial através de custo ou prazo é extremamente difícil. Levando em consideração que a mão de obra desse setor que possui uma qualificação bem nivelada, onde profissionais têm quase um mesmo nível de aptidão técnica resultando em um custo de mão de obra e prazo bem equilibrado entre as empresas, a qualidade desponta como um grande diferencial. Obter um processo de qualidade que promove uma cultura organizacional de garantia e controle da qualidade dos seus produtos torna-se essencial. As empresas que alcançam um nível de maturidade de processo de qualidade, certificado através de instituições independentes, reconhecidas pelo mercado, possui um grande diferencial. Empresas com esse tipo de certificado podem usufruir de um processo seguro, eficaz e controlado, capaz de disseminar uma cultura organizacional para gestão racional do desenvolvimento de seus produtos de software, além explorarem como apelo comercial.

A qualidade de software é um processo sistemático que focaliza todas as etapas e artefatos produzidos com o objetivo de garantir conformidade de processos e produtos prevenindo e eliminando defeitos (BARTIÉ, 2002). De acordo com BARTIÉ é impossível obter um software de qualidade com processos de desenvolvimento frágeis e deficientes. Vemos então que qualidade de software está intrinsecamente ligada a qualidade dos processos de produção deste produto. Podemos estabelecer então duas dimensões fundamentais da qualidade do software: qualidade de processo e qualidade do produto. Diante de um desafio de

garantir a qualidade de um software, de fato busca-se uma cultura de inibição de falhas e erros, estruturando processos que possuam mecanismos para mitigar brechas de possíveis fontes de defeitos (BARTIÉ, 2002). Esses processos estruturados buscam avaliar a qualidade de todas as saídas geradas durante o ciclo de desenvolvimento através de procedimentos e métricas. Dessa forma todo o processo de desenvolvimento deste produto estará coberto e garantido. Portanto, para se alcançar a qualidade de software é preciso que a garantia da qualidade de software seja parte integrante do ciclo de desenvolvimento e esteja presente em todas as fases.

A Garantia da qualidade de Software, *Software Quality Assurance* - SQA, deve ser um processo gerido de forma independente do processo de desenvolvimento do produto e se possível deve estabelecer uma relação de "um-para-um" entre as fases de desenvolvimento e as atividades a serem desempenhas por uma equipe de qualidade (BARTIÉ, 2002). Para PRESSMAN (PRESSMAN, 1995), a garantia da qualidade de software é uma “atividade de guarda-chuva” que é aplicada ao longo de todo o processo de engenharia de software. Isso nada mais é que, para se garantir a qualidade é esperado que haja para cada etapa de produção, um processo de avaliação da aderência e conformidade da saída produzida e dos processos, obtendo assim ao final de cada iteração um resultado com mais qualidade em sua totalidade, ou seja, diversos pedaços com qualidade remontam um todo com mais qualidade.

Partimos então deste ponto para tentar descrever o que é Garantia de Qualidade de Software. Dentre os mais notáveis trabalhos sobre engenharia de software (PRESSMAN, 1995), temos sobre garantia da qualidade de software uma abordagem que abrange:

- 01: métodos e ferramentas de análise, projeto, codificação e teste;
- 02: revisões técnicas formais que são aplicadas durante cada fase de engenharia de software;
- 03: uma estratégia de teste de múltiplas fases;
- 04: controle da documentação de software e das mudanças feitas nela;

05: um procedimento para garantir a adequação aos padrões de desenvolvimento de software (quando aplicáveis);

06: mecanismo de medição e divulgação.

Podemos observar que PRESSMAN prevê um conjunto robusto de processos para que o produto de software esteja em conformidade com os padrões de qualidade. Assim como um produto de manufatura, um produto de software com qualidade pode ser entendido como algo que possui uma adesão estrita e consistente com os padrões mensuráveis e verificáveis para alcançar a uniformidade da produção que satisfaça os requisitos específicos do cliente ou do usuário.

Invariavelmente a garantia da qualidade de software está atrelada a verificação e validação através de modelos de gerenciamento e de controle, como *Capability Maturity Model Integration* (CMMI), ISO 9000-3, ISO/IEC 15504, ISO/IEC 25051 e MPS.BR (MPS-SW) que foram motivados pelas falhas nos processos de gerência e manutenção durante o desenvolvimento de software (CESAR, 1997) e ainda a antiga ISO/IEC 9126 revisada pela ISO/IEC 25010:2011 para avaliação da qualidade do produto de software através de um conjunto de características (BARBACCI, 1995). Dessa forma definimos que as duas dimensões de qualidade de software: qualidade do processo e qualidade do produto estão amplamente cobertas por modelos difundidos no mercado, desenvolvidos e amadurecidos ao longo dos anos através de pesquisa e desenvolvimento prático.

O objetivo deste trabalho é apresentar uma visão geral sobre o processo de garantia da qualidade de software propondo um guia geral para estabelecer um conjunto de papéis e atividades com o objetivo de fornecer um guia para as empresas que buscam estabelecer qualidade nos processos de construção de software. Será descrito com ênfase o modelo CMMI-DEV, considerado um dos melhores modelos de qualidade para empresas de produtos e serviços, sendo também utilizado como base para o método proposto.

## **2. QUALIDADE**

### **2.1. Conceito**

Qualidade é um conceito subjetivo e está relacionado às percepções de cada indivíduo sob os diversos fatores como: cultura, crenças, valores, posição social e entre outros. Em termos de produtos e serviços qualidade se relaciona diretamente com as exigências, expectativas e necessidades dos clientes, como por exemplo: valor agregado, relação custo-benefício, durabilidade e acessibilidade.

Seguindo nesta direção temos algumas definições importantes já conhecidas e estudadas. "Qualidade é a habilidade de um conjunto de características inerentes a um produto, componente de produto ou processo atenderem aos requisitos dos clientes" (SEI 2006). Já para a norma ISO 8402 (ABNT, 1994), "Qualidade é a totalidade das características de uma entidade, que lhe confere a capacidade de satisfazer às necessidades explícitas e implícitas dos *stakeholders*".

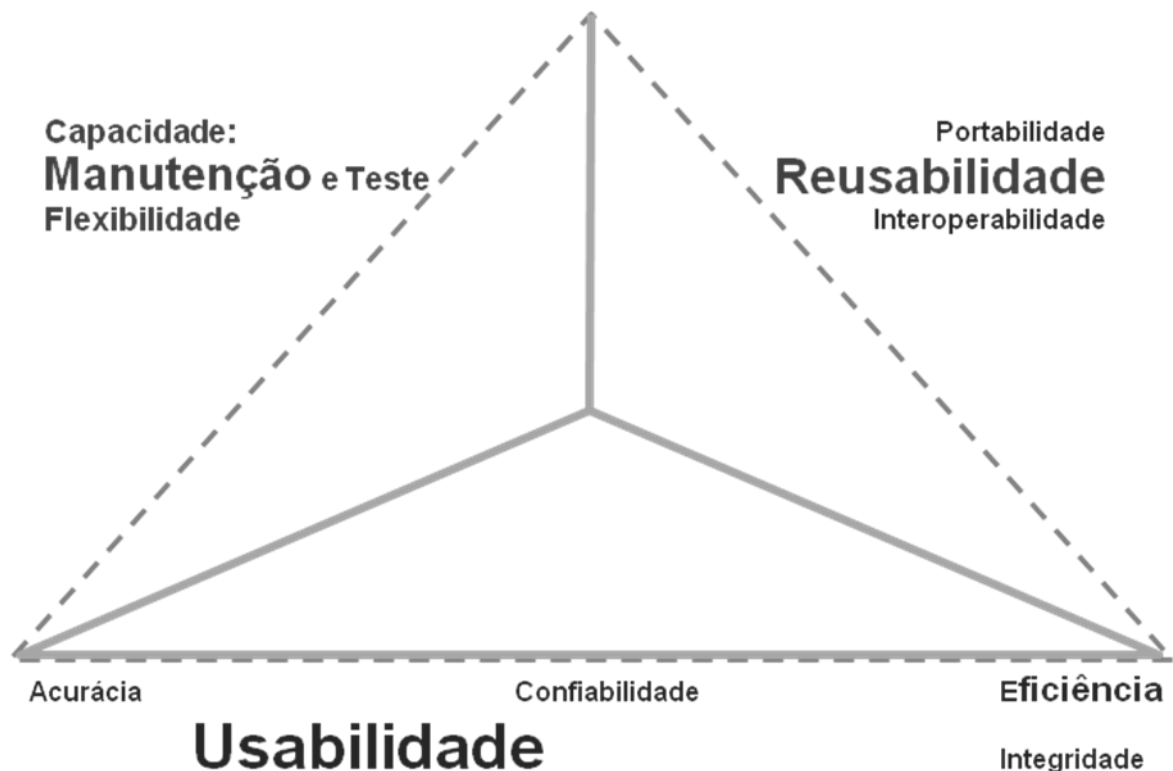
Podemos observar que qualidade está diretamente vinculada com as expectativas e necessidades do cliente. Segundo Philip Crosby (CROSBY, 1979) qualidade é estabelecer conformidade com os requisitos. Para a norma ISO9000 estabelece também que a "qualidade é o grau no qual um conjunto de características inerentes satisfaz a requisitos" (ISO9000 2005).

Desta forma, assim como para outros setores, qualidade é fator crítico de sucesso para a indústria de software (MR-MPS-SW:2012, 2012), tendo em vista o acelerado avanço tecnológico rápido e inovador e também cada vez mais presente no cotidiano dos indivíduos.

### **2.2. Qualidade de Software**

A Qualidade de Software é também algo complexo, no entanto, alguns modelos ou conjunto de atributos podem classificar a qualidade. McCall (MACCALL, 1977) propõe uma categorização dos fatores da qualidade de software, com ênfase em três

aspectos importantes: Características manutenção, operação, e sua capacidade de adaptabilidade.



**Figura 1. Modelo McCall (1977)**

Boehm acrescentou algumas características com o modelo de McCall com ênfase na capacidade de manutenção do produto de software (BOTELLA, 2004). Além disso, o modelo inclui considerações envolvidas na avaliação de um software produto no que diz respeito à utilidade do programa. O modelo proposto por Boehm é similar ao modelo de McCall e que representa uma hierárquica estrutura de características, cada um dos quais contribui para a qualidade total. A noção de Boehm inclui as necessidades dos usuários, assim como McCall, no entanto, o modelo de Boehm contém apenas um diagrama, sem qualquer sugestão de como medir as características de qualidade (BEHKAMAL, 2008).

Outro modelo também muito conhecido foi o FURPS (WATSON, 2006) proposto por Robert Grady e HewlettPackard Co., que decompõe características em duas categorias diferentes de requisitos:

Os requisitos funcionais: Definido pela entrada e saída esperada.

Os requisitos não funcionais: Usabilidade, confiabilidade, desempenho e capacidade de suporte.

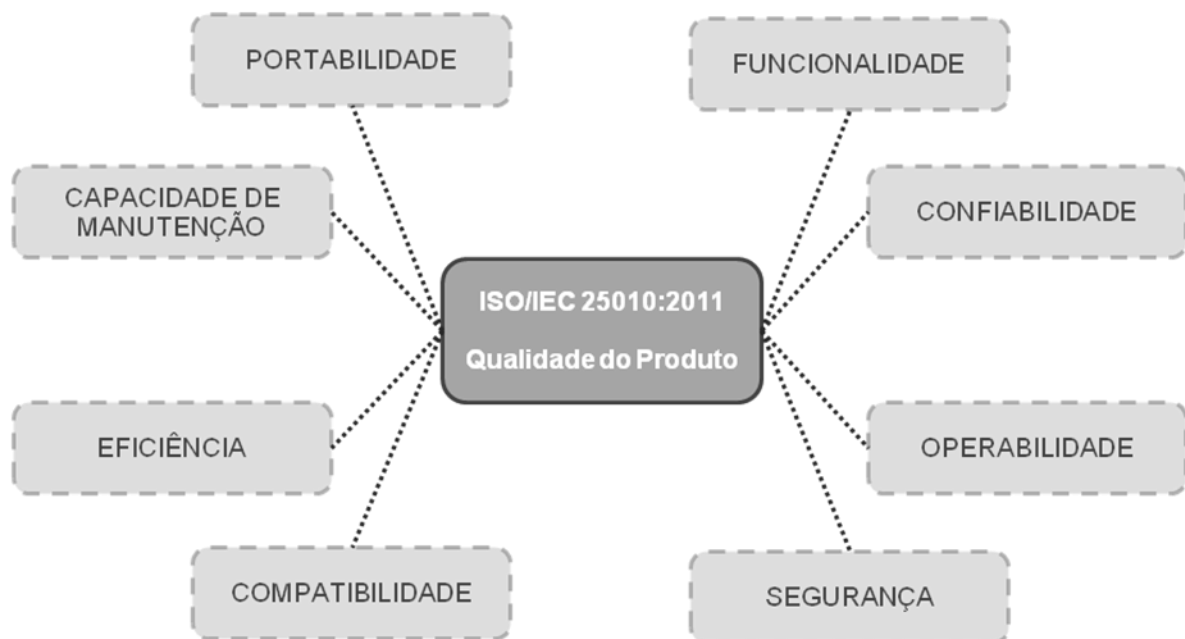
Uma desvantagem do modelo FURPS é que ele não leva em conta a portabilidade do produto de software (BEHKAMAL, 2008).

Muitos outros modelos surgiram, contudo, havia a necessidade de um modelo padrão, e com isso deu-se início a ISO/IEC JTC1 que buscou desenvolver consensos necessários e incentivar a normalização a nível mundial sobre os modelos de qualidade de software. As primeiras considerações foram feitas em 1978, e dando início em 1985 a ISO/IEC 9126. A ISO/IEC 9126 é parte da família ISO 9000, que é o padrão mais importante para a garantia da qualidade, no entanto, foi revisada em 2011 pela norma ISO/IEC 25010:2011 (ISO/IEC 25010, 2011), também conhecida como **SQuaRE** (*Software product Quality Requirements and Evaluation*).

A norma ISO/IEC 25010 propõe um conjunto de atributos de qualidade, distribuídos em 08 (oito) conjuntos principais, com cada um deles divididos em subcaracterísticas descritas pela norma. A nova norma tem oito conjuntos de características de qualidade do produto, em contraste com a ISO 9126 com seis, e 31 subcaracterísticas.

- Funcionalidade: A capacidade de um software prover funcionalidades que satisfaçam o usuário em suas necessidades declaradas e implícitas, dentro de um determinado contexto de uso.
- Confiabilidade: O produto se mantém no nível de desempenho nas condições estabelecidas.
- Operabilidade: Observa a adequação, fácil utilização, não exposição de erros ao usuário, cuidado com a estética da interface com o usuário e acessibilidade,

- Segurança: Preocupação com confidencialidade, integridade, responsabilidade e autenticidade.
- Compatibilidade: coexistência e interoperabilidade.
- Eficiência: O tempo de execução e os recursos envolvidos são compatíveis com o nível de desempenho do software.
- Capacidade de manutenção: A capacidade (ou facilidade) do produto de software ser modificado, incluindo tanto as melhorias ou extensões de funcionalidade quanto às correções de defeitos, falhas ou erros.
- Portabilidade: A capacidade de o sistema ser transferido de um ambiente para outro.



**Figura 2. Adaptado de ISO/IEC 25010 (2011)**

Podemos observar que temos a partir da ISO/IEC 25010 um conjunto bem amadurecido e suficientemente capaz que cobrir os diversos modelos já propostos de forma integrada.



Ainda assim podemos dizer que existem duas visões de qualidade de software, uma dos clientes e outra dos que desenvolvem o software, mas ambas concordam que o software não pode ter defeitos. O cliente avalia o software sem conhecer seus aspectos internos, está apenas interessado na facilidade do uso, no desempenho, na confiabilidade dos resultados obtidos e também no preço do software. Os que desenvolvem o software avaliam aspectos internos como taxa de defeitos, confiabilidade, facilidade de manutenção e também aspectos de conformidade em relação aos requisitos dos clientes (ISO, 1991), (CAROSIA, 2003). Para PRESSMAN (PRESSMAN, 1995) essas duas visões podem ser exemplificadas como:

1. Fatores que podem ser medidos diretamente (por exemplo, erros/KLOC/unidade de tempo)
2. Fatores que podem ser medidos apenas indiretamente (por exemplo, usabilidade ou capacidade de manutenção).

### **2.3. Qualidade de processo**

Antes de discutir sobre qualidade do processo, devemos definir o que é entendido como processo. Processo são um conjunto de passos ordenados, constituídos por atividades, métodos, práticas e transformações, usado para atingir uma meta. Esta meta geralmente está associada a um ou mais resultados concretos finais, que são os produtos da execução do processo (PÁDUA, 2003).

Um processo de software é um conjunto de ferramentas, métodos e práticas usadas para produzir software. O processo de software é representado por um conjunto sequencial de atividades, objetivos, transformações e eventos que integram estratégias para cumprimento da evolução de software (PRESSMAN, 1995). A norma ISO9000 define processo como um conjunto de atividades inter-relacionadas ou interativas que transformam entradas em saídas.

Como a qualidade do produto pode ser vista como uma consequência da qualidade do processo, os certificados mais valiosos são aqueles que certificam o

processo de produção de um produto e não os que certificam simplesmente o produto. Os estudos sobre qualidade, mais recentemente, são voltados para o melhoramento do processo de software, pois ao garantir a qualidade do processo, já se está dando um grande passo para garantir também a qualidade do produto (CAROSIA, 2003).

Um processo de qualidade, de maneira geral, é ter um processo documentado, estabelecidos a partir de modelos amadurecidos, e que buscam a melhoria contínua. Os principais modelos da atualidade são o *Capability Maturity Model Integration* (CMMI) e a norma ISO/IEC 15504-5. No cenário nacional, o Modelo de Referência MPS que faz parte do MPS.BR (Melhoria de Processo do Software Brasileiro).

## **2.4. Os pilares da qualidade**

Para que haja efetivamente um produto ou serviço de software de qualidade é proposto um processo de gerenciamento. O processo de gerenciamento da qualidade que é subdividido em três subprocessos complementares: Planejamento da qualidade, Garantia da qualidade e Controle da qualidade, seguindo os trabalhos realizados pelo PMI (*Project Management Institute*) e os organizou no **PMBOK** (*Project Management Body of Knowledge*) (BARTIÉ, 2002).

### **2.4.1. Planejamento da qualidade**

Processo destinado a identificar quais padrões de qualidade é relevante para o projeto e determinar como satisfê-los. É realizado em paralelo com outros processos de planejamento e tem como produto o Plano da Garantia da Qualidade de Software (*SQA Plan - Software Quality Assurance Plan*) e todos os planejamentos mais direcionados (Estratégias de Testes das diversas categorias existentes) (BARTIÉ, 2002).

#### **2.4.2. Garantia da qualidade**

Processo que engloba a estruturação, sistematização e execução das atividades que terão como objetivo garantir o adequado desempenho de cada etapa do desenvolvimento (BARTIÉ, 2002).

#### **2.4.3. Controle da qualidade**

O processo que se concentra no monitoramento e desempenho dos resultados do projeto para determinar se ele está atendendo aos padrões de qualidade no processo de desenvolvimento (BARTIÉ, 2002).

Para fins de desambiguação, cabe diferenciar o conceito de garantia da qualidade do conceito de controle da qualidade. O controle da qualidade é uma atividade inserida em processo de software que garante o pleno funcionamento do produto ou serviço de acordo com os requisitos previamente levantados (BASTOS, 2007), enquanto que garantia da qualidade remete ao como fazer o software. Por outro lado, o controle da qualidade é a comparação de um determinado produto ou serviço com suas respectivas especificações, onde todo e qualquer desvio identificado é devidamente registrado e endereçado.

### **3. MODELO DE MATURIDADE CMMI (*Capability Maturity Model Integration*)**

#### **3.1. CMM**

Antes de descrever sobre o modelo CMMI, é extremamente importante colocar seu contexto histórico e motivação. O CMM (*Capability Maturity Model*), como descrito a seguir, é o precursor do modelo CMMI, motivados pela necessidade de se ter sob o controle processos de software e uma padronização no processo de desenvolvimento durante um período descrito como a crise de software.

O CMM (PAULK, 1993) foi desenvolvido pelo SEI (*Software Engineering Institute*), ligado a Universidade Carnegie Mellon e financiado pelo Departamento de Defesa Norte-Americano, com o objetivo de estabelecer um padrão de qualidade para o software desenvolvido para as Forças Armadas. Este modelo foi proposto por Watts S. Humphrey, a partir de conceitos de qualidade total estabelecido por Crosby. Segundo Tingey (1997), Crosby mostrou que a implementação de sistemas de qualidade em empresas segue um amadurecimento gradativo em patamares denominados “incerteza, despertar, esclarecimento, sabedoria e certeza”.

Em 1987, a partir do trabalho de Humphrey, o SEI lançou uma breve descrição de um ambiente de maturidade de processo de software e um questionário de maturidade para avaliar o estado corrente das práticas de software e identificar as áreas que necessitavam de melhoria. Em 1991, o SEI criou o CMM, como uma evolução deste ambiente. A partir do lançamento e divulgação da versão 1.1 do CMM, em 1993, o tema de melhoria do processo foi ganhando força. Esta força foi consequência dos resultados práticos obtidos pelas organizações que realizaram programas de melhoria com o CMM como modelo de referência (GONÇALVES, 2001), (CAROSIA, 2003 p.34).

Desde sua primeira versão, lançada em 1991, na esteira de sucesso do SW-CMM, diversos outros modelos foram criados visando cobrir outras áreas de interesse, como por exemplo:

- SA-CMM - *Software Acquisition CMM*: usado para avaliar a maturidade de uma organização em seus processos de seleção, compra e instalação de software desenvolvido por terceiros.
- SE-CMM - *System Engineering CMM*: avalia a maturidade da organização em seus processos de engenharia de sistemas, concebidos como algo maior que o software. Um sistema inclui o software, o hardware e quaisquer outros elementos que participam do produto completo.
- IPD-CMM - *Integrated Product Development CMM*: ainda mais abrangente que o SE-CMM, inclui também outros processos necessários à produção e suporte ao produto, tais como suporte ao cliente e processos de fabricação.
- P-CMM - *People CMM*: avalia a maturidade da organização em seus processos de administração de recursos humanos no que se refere a software, recrutamento e seleção de profissionais e treinamento.

O surgimento de todos estes outros modelos acabou trazendo alguns problemas. Como nem todos os modelos usavam a mesma terminologia, um mesmo conceito podia receber nomes diferentes em cada modelo. Os modelos tinham diferentes números de níveis ou formas diferentes de avaliar o processo. Outro problema era os altos custos de treinamento, avaliação e harmonização para organizações que quisessem usar mais de um modelo. Assim, tornou-se necessária uma padronização. Além disso, a experiência no uso de SW-CMM, durante uma década, serviu para identificar pontos em que o modelo poderia ser melhorado. O surgimento do projeto SPICE levou à necessidade de tornar o CMM compatível com a norma ISO 15504 (1997), resultado deste projeto.

Por estas razões, o SEI iniciou um projeto chamado CMMI - *CMM Integration*. Seu objetivo era gerar uma nova versão do CMM que resolvesse esses problemas. Concretamente, a primeira ideia, como o nome sugere, foi integrar os diversos CMMs em uma estrutura única, todos com a mesma terminologia, processos de avaliação e estrutura. O projeto se preocupou ainda em tornar o CMM compatível com a norma ISO/IEC 15504, de modo que avaliações em um modelo sejam reconhecidas como equivalentes aos do outro. E, naturalmente, incorporar ao CMM as sugestões de melhoria surgidas ao longo dos anos.

O Modelo de Maturidade da Capacidade Integrado, ou CMMI, foi resultado da integração dos modelos (CAROSIA, 2003):

- SW-CMM V2C - *Capability Maturity Model for Software V2.0, draft C.*
- SECM - *EIA Interim Standard 731 – System Engineering Capability Model.*
- IPD-CMM - *Integrated Product Development Capability Maturity Model, draft 0.98.*

O CMMI é claramente uma melhoria sobre o CMM. No entanto, apesar das melhorias incorporadas ao CMMI, seu precursor - o CMM - continua sendo um modelo bastante eficiente, (CAROSIA, 2003 p.41e 42).

### 3.2. CMMI

O CMMI, que é o sucessor do modelo de maturidade (CMM) ou CMM Software, como já foi descrito. O CMMI foi desenvolvido no período de 1987 até 1997. Em 2002, foi lançada a versão 1.1 do CMMI, seguido da versão 1.2 em agosto de 2006, e CMMI versão 1.3 em novembro de 2010. Algumas das principais mudanças no CMMI versão 1.3 é o apoio de *Agile Software Development* (Desenvolvimento ágil de Software), e melhorias nas práticas de maturidade, assim como, o alinhamento da representação (estágio e contínua).

**Representação Contínua:** Mais flexível, porém mais complexa de administrar. Permite a seleção da ordem de melhoria dos processos que melhor se adéqua aos objetivos de negócio da organização, além de permitir que sejam feitas comparações entre áreas de processo entre diferentes organizações ou através dos resultados apresentados de acordo com a equivalência de estágios. Possui uma estrutura compatível com padrão ISO/IEC 15504.

**Representação por Estágios:** organiza as áreas de processo em cinco níveis de maturidade. Esses níveis de maturidade servem para dar suporte e orientar a melhoria do processo, indicando quais áreas implementar para atingir cada nível.

O CMMI é originado em engenharia de software, mas tem sido muito generalizado ao longo dos anos para abraçar outras áreas de interesse, tais como o

desenvolvimento de produtos de hardware, a entrega de todos os tipos de serviços e aquisição de produtos e serviços. A palavra "software" não aparece nas definições de CMMI. Esta generalização de conceitos de melhoria faz do CMMI extremamente abstrato. Não é tão específico para engenharia de software como o seu antecessor, o Software CMM. O CMMI atualmente aborda três áreas de interesse:

Produtos e serviços de desenvolvimento - CMMI *for development* (CMMI-DEV);

Gestão de estabelecimento de serviços - CMMI *for Services* (CMMI-SVC);

Aquisição de produtos e serviços - CMMI *for Acquisition* (CMMI-ACQ).

O modelo com maior aderência para empresas de desenvolvimento de software é o CMMI-DEV, pois suporta duas representações: contínua e em estágios, sendo considerado também um modelo flexível. A primeira representação é voltada para avaliação e definição do nível de capacidade de processos escolhidos para serem avaliados em uma determinada unidade organizacional, enquanto a segunda representação é voltada à avaliação de um conjunto de processos pré-definidos em cada nível de maturidade do modelo CMMI (SEI, 2006).

### **3.3. CMMI-DEV (*for development*)**

O modelo CMMI-DEV V1.3 (CMMI, 2010) é um conjunto de melhores práticas de desenvolvimento usado pelo governo dos Estados Unidos da América e pela indústria que é gerado a partir do CMMI V1.3 Arquitetura e *Framework*. O *Framework* CMMI é a estrutura básica que organiza componentes CMMI e as combina em constelações e modelos CMMI. Uma constelação é uma coleção de componentes CMMI que são usados para construir modelos, materiais de treinamento e documentos de avaliação relacionada para uma área de interesse (por exemplo, desenvolvimento, aquisição, serviços).

O CMMI *for development* fornece orientação necessária para aplicar as melhores práticas do CMMI em uma organização, para se concentrar em atividades de desenvolvimento de produtos e serviços de qualidade para atender às necessidades dos clientes e usuários finais.

Este modelo é baseado na CMMI *Model Foundation* ou CMF (ou seja, os componentes do modelo comuns a todos os modelos CMMI e constelações) e incorpora o trabalho das organizações de desenvolvimento para adaptar CMMI para uso no desenvolvimento de produtos e serviços.

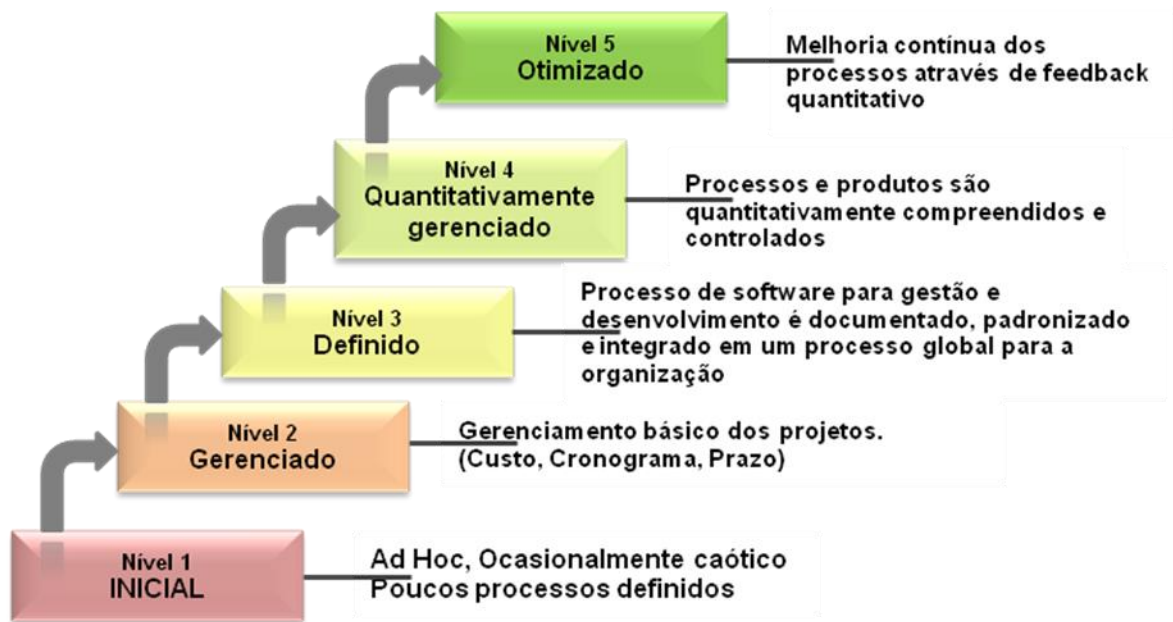
Ele contém 22 áreas de processo. Dessas áreas de processos, 16 são áreas de processo do núcleo (*core*), 01 é uma área de processo compartilhado e 05 são áreas específicas do processo de desenvolvimento. Uma área de processo principal é uma área de processo que é comum a todos os modelos CMMI. Uma área de processo comum é compartilhada por pelo menos dois modelos CMMI, mas não todas elas.

É um modelo de referência que abrange as atividades para o desenvolvimento de produtos e serviços em diversas organizações do setor industrial, incluindo aeroespacial, bancário, hardware, software, de defesa, indústria automobilística e de telecomunicações e contém práticas que abrangem gestão de projetos, gestão de processos, engenharia de sistemas, engenharia de hardware, engenharia de software, e de outros processos de apoio utilizadas no desenvolvimento e manutenção (CMMI, 2010).

### **3.3.1. Níveis de maturidades contidas no CMMI-DEV**

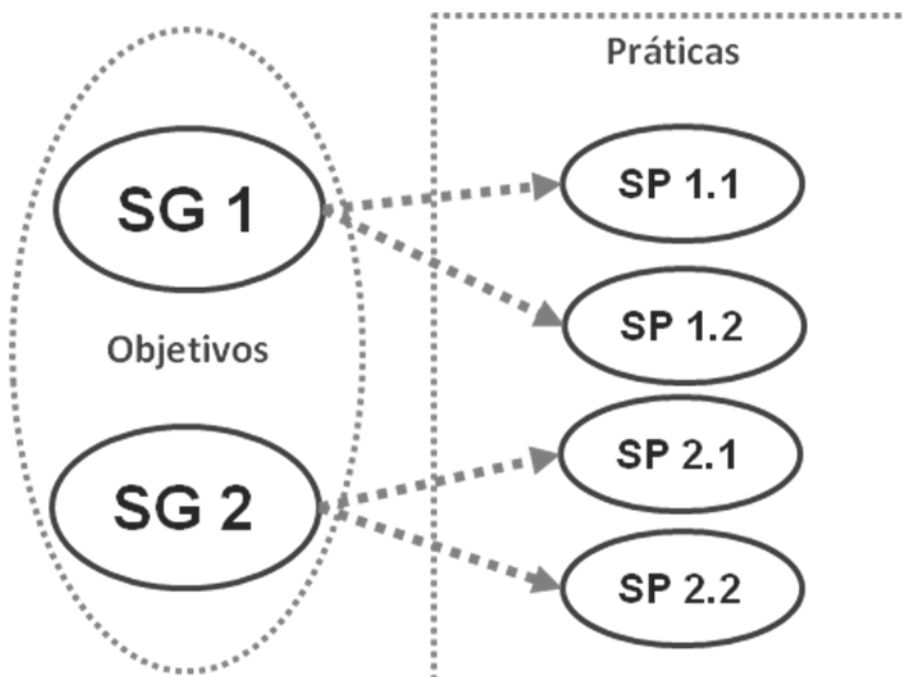
Antes de abordar a diretamente a área de processo de Garantia da Qualidade (PPQA) é importante apresentar de forma sucinta a estrutura do CMMI-DEV. No CMMI-DEV há 05 (cinco) níveis de maturidade. No entanto, as avaliações dos níveis de maturidade são concedidas para níveis de 02 a 05 (cinco). Cada nível de maturidade agrega um conjunto de áreas de processos que devem ser praticados para se alcançar o nível de maturidade.





**Figura 3. Níveis de maturidade do CMMI. Adaptado de CMMI (2010)**

Área de processo é um grupo de práticas relacionadas com um processo específico que quando executado satisfaz um conjunto de objetivos considerados importantes para a melhoria deste processo (SEI, 2006). As áreas de processo possuem componentes que possuem objetivos e práticas, específicas, genéricas ou informativas.



**Figura 4. Ilustração, um exemplo de área de processo. CMMI (2010).**

**Práticas Específicas – (SP *Specific Practices*):** atividades que são consideradas importantes na satisfação de uma meta específica associada.

**Práticas genéricas - (GP *Generics Practices*):** oferecem uma institucionalização que assegura que os processos associados com a PA serão eficientes, repetíveis e duráveis.

**Objetivos Genéricos - (GG *Generic Goals*):** Compartilhado em diversas áreas de processo.

**Objetivos específicos – (SG *Specific Goals*):** Objetivos específicos a uma área de processo

**Informativas:** Produtos de trabalho típicos, exemplos de saídas de uma prática específica ou genérica. Subpráticas: descrições detalhadas que fornecem um direcionamento para a interpretação de práticas específicas ou genéricas.

Não será descrito neste trabalho, as práticas e objetivos genéricos a todas as áreas, tendo em vista que há uma grande e vasta descrição em detalhes presente na literatura (GONÇALVES, 2001), (CAROSIA, 2003), (CMMI, 2010), (CHRISSIS, 2011) e não é o objetivo central deste trabalho.

## ÁREAS DE PROCESSOS AGRUPADAS POR NÍVEL DE MATURIDADE

Nível	Sigla em Inglês	Descrição
2	CM	Gestão da configuração
	MA	Medição e Análise
	PMC	Projeto de Monitoramento e Controle
	PP	Planejamento de Projetos
	<b>PPQA</b>	<b>Garantia da qualidade do Processo e Produto</b>
	REQM	Gerenciamento de Requisitos
	SAM	Gerenciamento de acordo com o fornecedor
3	DAR	Análise de decisão e resolução
	IPM	Gestão Integrada de Projetos
	OPD	Definição do Processo Organizacional
	OPF	Foco no Processo Organizacional
	OT	Treinamento Organizacional
	PI	Integração de Produto
	RD	Desenvolvimento de Requisitos
	RSKM	Gestão de Riscos
	TS	Solução Técnica
	VAL	Validação
	VER	Verificação
4	OPP	Desempenho do Processo Organizacional
	QPM	Projeto quantitativamente gerenciado
5	CAR	Análise e Resolução de Causas
	OPM	Gestão de Desempenho Organizacional

**Tabela 1. Áreas de processo agrupadas por nível. Adaptado de CMMI (2010)**

### **3.3.2. Garantia da qualidade do produto e processo (PPQA).**

A área de processo de garantia da qualidade do produto e processo do CMMI-DEV, (*Process and Product Quality Assurance* - PPQA) está contido dentro do nível 02 de maturidade (**ver Tabela 1**), ou seja, o primeiro nível a ser buscado pelas organizações que esperam estabelecer uma metodologia seguindo este modelo.

O objetivo do grupo de processo da Garantia da qualidade do Processo e Produto (PPQA) é fornecer a equipe e gestores uma visão objetiva dos processos e produtos de trabalho associados (CHRISSIS, 2011). Abaixo segue a descrição extraída do CMMI-DEV, referentes aos objetivos e praticas específica desta área de processo.

---

#### **SG 1 - Aderência dos processos e produtos de trabalhos associados com as descrições do processo, padronizar e avaliar objetivamente o processo.**

SP 1.1 Avaliar objetivamente processos realizados, selecionado contra descrições aplicáveis de processo, padrões e procedimentos.

SP 1.2 Avaliar objetivamente produtos de trabalho selecionado contra as descrições aplicáveis de processo, padrões e procedimentos.

#### **SG 2 - Questões de não conformidades são objetivamente rastreadas e comunicados, e é assegurada a resolução.**

SP 2.1 Comunicar problemas de qualidade e garantir a resolução dos problemas de não conformidade com a equipe e gestores.

SP 2.2 Estabelecer e manter registros das atividades de garantia de qualidade.

---

#### 4. VISÃO GERAL PARA ESTABELEECER O PPQA

Neste trabalho, não será apresentado um guia detalhado para estabelecer os processos e práticas de garantia da qualidade de processo e produto, requeridos para estar conformidade com a certificação junto ao SEI, (CMMI-DEV, Nível 02), tendo em vista que para isto, seria de vital importância um estudo de caso ou ainda conhecer modelos amplamente estabelecidas por organizações já certificadas.

No entanto, será proposta uma estrutura de papéis e atividades, para que se tenha um guia inicial, uma visão geral. Vale a pena citar alguns modelos publicados, como o *framework* proposto por (ELBERZHAGER, 2007) e também o modelo de processos de PPQA publicado por (MANUVANNAN, 2010), que estabelece um conjunto de processos e artefatos que devem estar contidos na abordagem de PPQA.

De modo geral, estes modelos propõe a criação de um órgão ou departamento com papéis e responsabilidade independente da área de produção ou desenvolvimento do produto a fim de executar um conjunto de processos estabelecidos pela organização para que esteja em conformidade com as práticas específicas contidas na área de processo PPQA do CMMI-DEV (CMMI, 2010), (MANUVANNAN, 2010) e (ELBERZHAGER, 2007). O modelo aqui proposto, parte também da premissa que as atividades de garantia da qualidade estão envolvidas em um ciclo de projeto, independente do ciclo de vida adotado para o projeto.

De forma, clara e resumida PPQA envolve as seguintes questões que devem estar em foco a cada atividade:

- Avaliar a aderência dos produtos de trabalho e serviços com as descrições dos processos, padrões e procedimentos estabelecidos.
- Identificar e documentar problemas de não conformidade.
- Manter equipe e gestores informados dos resultados sobre as atividades da Garantia da Qualidade.
- Certificar que problemas de não conformidade estão devidamente delegados (MANUVANNAN, 2010).

Dessa forma vamos estabelecer um conjunto de artefatos e tarefas de acordo com os trabalhos já citados e alguma experiência em vivência com esse modelo.

#### **4.1. Artefatos**

Os artefatos podem ser classificados em 02 (dois) tipos. Entrada e Saída (MANUVANNAN, 2010). Os artefatos de entradas podem ser considerados documentos que geram uma base ou orientação para as atividades de PPQA. Já os artefatos de saídas serão considerados documentos que serão produzidos pelas atividades de PPQA, ou seja, resultado de trabalhos da Garantia da Qualidade.

Uma observação importante é que as siglas definidas logo após a descrição do artefato, não segue nenhum padrão estabelecido ou proposto e é meramente com o intuito de organização deste trabalho.

##### **Entrada:**

- Plano de projeto (PRPL)
- Documentação de processos (PDOC)
- *Checklist* (P-CHK)

##### **Saída:**

- Plano de qualidade (QAPL)
- Relatório periódico da Garantia da Qualidade (RP-STQA)
- Relatório de não conformidades (RP-NC)
- Relatório sobre lições aprendidas (RP-LL)

#### **4.2. Atividades**

As atividades de garantia da qualidade são estruturadas para atender as demandas da empresa, contudo, com o foco em atender as avaliações do SEI. Mais adiante será feita uma análise de aderência do modelo proposto com os objetivos e práticas específicas da área de PPQA do CMMI-DEV v 1.3.

1. Disponibilizar materiais de treinamento para os integrantes do projeto
2. Elaborar plano de qualidade

3. Avaliar objetivamente os produtos de trabalho e serviços
4. Avaliar objetivamente os processos
5. Elaborar relatório de não conformidades
6. Elaborar relatório periódico de garantia da qualidade
7. Aprovar produto final para entrega
8. Elaborar relatório de lições aprendidas do projeto
9. Prestar suporte a todos os envolvidos no projeto

Todas essas atividades estão distribuídas entre os papéis previstos. Tendo em vista que cada papel, neste caso representa uma função a ser desempenhada não determinando cargo, hierarquia nem o número de pessoas para desempenhar cada papel. Isso cabe a cada organização definir e estabelecer de acordo com a sua estrutura organizacional, necessidades e objetivos.

---

### PAPÉIS PREVISTOS NA ÁREA DE GARANTIA DA QUALIDADE

---

Sigla em Inglês		Nome
	PM	Gerente de projetos
	SM-QA	Gerente sênior em garantia da qualidade
	QA-R	Representante da garantia da qualidade
	MI	Responsável pelas medições e métricas

**Tabela 2. Papéis previstos. Adaptado de MANUVANNAN (2010).**

Para que as atividades descritas sejam desempenhadas, alguns passos serão propostos a seguir para obter um nível um pouco mais detalhado e alguns exemplos práticos.

#### **1. Disponibilizar materiais de treinamento para os integrantes do projeto**

- a. Desenvolver conteúdo de treinamento para os integrantes do projeto sobre Garantia da qualidade e atividades de PPQA.
  - Em geral, uma boa opção é disponibilizar esse conteúdo através de uma Intranet.
- b. Desenvolver questionário para avaliar o aprendizado sobre PPQA

- Um pequeno questionário, também disponibilizado através de uma Intranet pode atender essa necessidade de maneira ágil.

## **2. Elaborar plano de qualidade (QAPL)**

- a. Identificar e entender o escopo do projeto.
  - Designar um representante de PPQA para atender a reunião de pontapé inicial do projeto.
- b. Identificar o modelo de ciclo de vida do projeto.
  - Analisar junto com o Gerente de projetos o tipo do projeto, a dimensão, as entregas previstas e lições aprendidas de projetos similares.
- c. Desenvolver o Plano de garantia da qualidade de processo e produto.
  - Eleger processos a serem avaliados;
  - Eleger produtos de trabalho a serem avaliados;
  - Estabelecer critério de avaliação. Este pode ser considerado um ponto crítico, pois a partir do critério estabelecido, que serão aplicadas das métricas nos processos e produtos de trabalho. O critério deve ser claro e objetivo.
  - Estabelecer o prazo e agendamento para as avaliações

## **3. Avaliar objetivamente os produtos de trabalho e serviços**

- a. Selecionar produtos de trabalho para serem avaliados com base no plano de qualidade.
- b. Realizar a avaliação do produto de trabalho.
- c. Coletar e documentar boas práticas e lições aprendidas que possam contribuir com a melhoria do produto, métodos e ferramentas utilizados.
- d. Identificar e documentar não conformidades encontradas.

## **4. Avaliar objetivamente os processos**

- a. Selecionar e detalhar os processos do projeto para ser avaliados
- b. Realizar avaliação a partir de critérios estabelecidos no plano de qualidade



- c. Coletar e documentar boas práticas e lições aprendidas que possam contribuir com a melhoria do processo.
- d. Identificar e documentar não conformidades encontradas.

## **5. Elaborar relatório de não conformidades (RP-NC)**

- a. Elaborar relatório de não conformidades encontradas na avaliação de processo e produto de trabalho.
  - Classificar as não conformidades com um grau de importância.  
**Alta:** Se afetar os produtos gerados pelo projeto.  
**Baixa:** Se não afetar os produtos gerados pelo projeto, exigir correções em documentos ou revisões de melhorias básicas.
- b. Descrever e delegar ações corretivas e assegurar a resolução.
  - Estabelecer um mecanismo de rastreamento das ações corretivas com o intuito de garantir que estão sendo seguidas.

## **6. Elaborar relatório periódico de garantia da qualidade (RP-STQA)**

- a. Elaborar e enviar para todos os envolvidos o relatório periódico.
  - O período deve ser acordado entre os envolvidos e o gerente de projeto. De forma geral um relatório semanal pode ser uma boa sugestão.
  - O relatório deve descrever as atividades realizadas pela Garantia da qualidade referente ao período em questão.

## **7. Aprovar produto final para entrega**

- a. Certificar que todos os processos foram seguidos em conformidade
- b. Em caso de não conformidades pendentes, garantir a rastreabilidade da mesma a fim de resolvê-las.
- c. Escalar ao nível competente em caso de não conformidades não tratadas devidamente.
- d. Aprovar o produto final para entrega.

## **8. Elaborar relatório de lições aprendidas do projeto (RP-LL)**

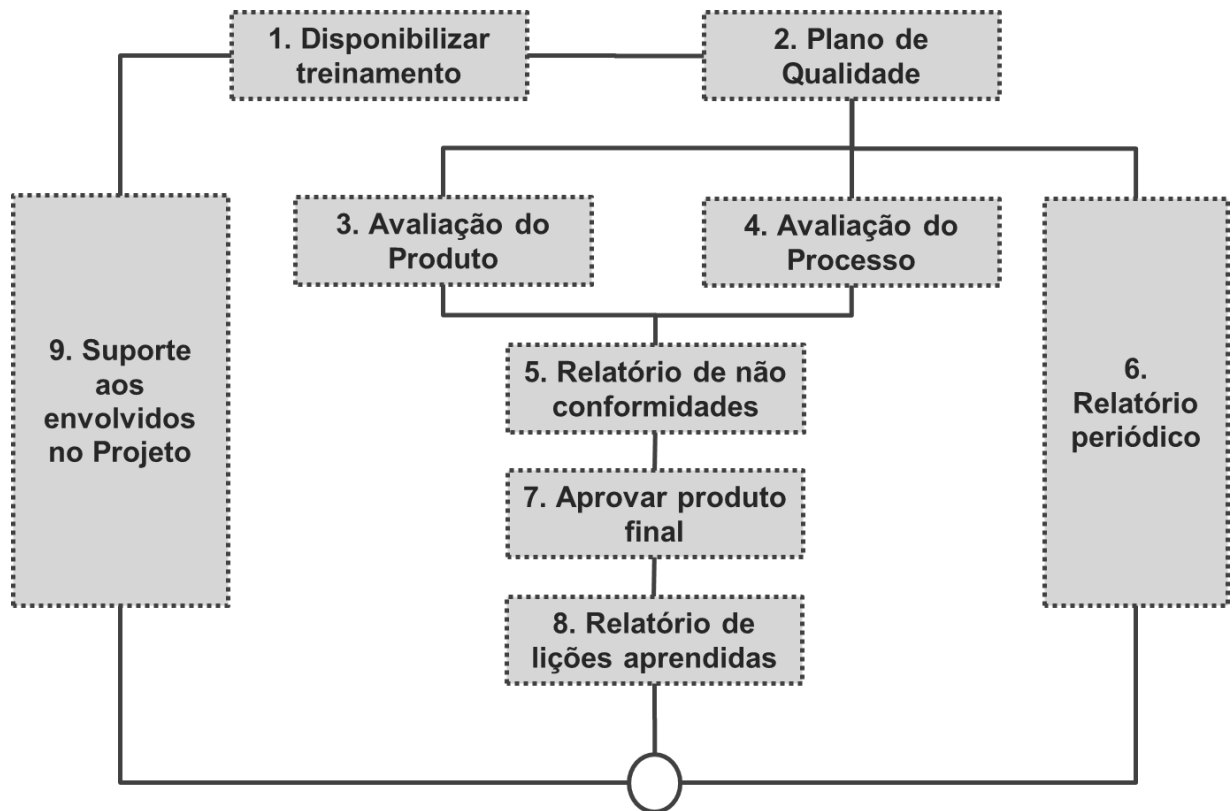
- a. Coletar, documentar e reportar lições aprendidas durante o projeto com o intuito de colaborar para projetos futuros.

- Identificar e incluir no relatório, boas práticas, observações e análises pessoais dos envolvidos no projeto.

## **9. Prestar suporte a todos os envolvidos no projeto**

- a. Designar um representante de PPQA para prestar suporte aos envolvidos no projeto.
- b. Estabelecer canais para contato.
  - Email, Telefone e *Chat* (Comunicador de texto ou voz instantâneo).

Podemos observar que algumas atividades geram insumos para outras, e vale lembrar que, não necessariamente seguem uma ordem de execução como enumeradas neste trabalho. Algumas atividades são recorrentes, outras assíncronas e outras ainda pontuais, no entanto, a Garantia da qualidade deve estar presente em todo o ciclo de desenvolvimento do projeto. Cabe aos integrantes de PPQA identifica-las e determinar como e quando executá-las buscando a melhor adequação aos processos já estabelecidos. Todavia, será representada a seguir uma estrutura, apenas como exemplo, das atividades propostas para uma melhor avaliação do resultado da proposta.



**Figura 5. Estrutura de atividades do método proposto**

Com o objetivo de avaliar a aderência das atividades propostas, a tabela a seguir ilustra a verificação de conformidade da proposta com o modelo CMMI-DEV v1.3. É estabelecido um paralelo com as práticas específicas de PPQA com o conjunto de atividades proposto, dessa forma é possível também estabelecer parâmetros para qualificar a adequação da proposta.

### Aderência do método proposto

CMMI-DEV - PPQA	Abordagem proposta
<b>SP 1.1</b> Avaliar objetivamente processos realizados selecionado contra descrições aplicáveis processo, padrões e procedimentos.	<b>4. Avaliar objetivamente os processos.</b>
<b>SP 1.2</b> Avaliar objetivamente produtos de trabalho selecionado contra as descrições aplicáveis processo, padrões e procedimentos.	<b>3. Avaliar objetivamente os produtos de trabalho e serviços.</b>
<b>SP 2.1</b> Comunicar problemas de qualidade e garantir a resolução dos problemas de não conformidade com a equipe e gestores.	<b>5. Elaborar relatório de não conformidades.</b>
<b>SP 2.2</b> Estabelecer e manter registros das atividades de garantia de qualidade.	<b>2. Elaborar plano de qualidade.</b>
	<b>6. Elaborar relatório periódico de garantia da qualidade.</b>
	<b>8. Elaborar relatório de lições aprendidas do projeto.</b>
<i>Extra - Atividade de Suporte e Informativas</i>	<b>1. Disponibilizar materiais de treinamento para os integrantes do projeto.</b>
	<b>7. Aprovar produto final para entrega.</b>
	<b>9. Prestar suporte a todos os envolvidos no projeto.</b>

**Tabela 3. Análise de aderência da proposta**

## 5. CONCLUSÃO

Neste trabalho, foi apresentada uma visão geral sobre o processo de garantia da qualidade de software. Com uma fundamentação teórica revisando os conceitos de qualidade, qualidade de software e qualidade de processo, assim como, uma descrição geral sobre alguns dos modelos mais conceituados que descrevem modelos de processos e conjunto de características pertinentes à qualidade de produtos e serviços de Software. Foi descrito com ênfase o modelo CMMI-DEV, considerado um dos melhores modelos de qualidade para empresas de produtos e serviços. Este modelo tem se mostrado cada vez mais eficiente, respeitado e cada vez mais explorado como base para outros modelos e abordagens propostas.

Com base na área de processo PPQA do CMMI-DEV, foi trazida uma análise e uma proposta de aderência para um conjunto de atividades como um guia geral para as organizações que buscam conformidade com o CMMI-DEV e um modelo de maturidade para garantir qualidade em seus produtos e, ou serviços de software.

### 5.1. Dificuldades encontradas

No decorrer do trabalho diversas dificuldades foram sendo superadas, e vale registrar para outros que possam se aventurar por este caminho.

- Falta de experiência sólida, ou vivência com o tema, tornou-se um desafio de entender e sugerir uma forma de alcançar a conformidade com o CMMI-DEV PPQA.
- Uma quantidade pequena de literaturas que propõe “como” estabelecer uma metodologia baseada em CMMI. O CMMI é um *framework*, ou seja, uma especificação e não uma “receita de bolo”, sendo assim e preciso “implementar” o *framework*.

### 5.2. Considerações finais

Podemos concluir que hoje existem diversos modelos e normas referentes à qualidade software (CMMI, 2010), (ISO/IEC 25000), (ISO/IEC 25010), (ISO/IEC 90003), (ISO/IEC 25041), (ISO/IEC 25050) e (MR-MPS-SW), no entanto, há uma

grande necessidade de modelos que propõe uma forma de estabelecer de forma prática, com descrições e exemplos, no intuito de auxiliar as organizações que buscam conformidades com tais modelos de referências.

Contudo, num setor onde a cada dia a competitividade está cada vez mais acirrada, entender e buscar a conformidade com um modelo que visa garantir qualidade e maturidade nos processos de desenvolvimento de software, que melhor se adéqua a necessidade e objetivos da organização, trarão benefícios como ganhos em produtividade, redução do custo ao médio e longo prazo, além de estabelecer uma cultura organizacional de aprimoramento contínuo dos processos.

### **5.3. Trabalhos futuros**

Uma sugestão natural para trabalhos futuros é detalhar as atividades a fim de descrever um guia mais completo e com exemplos práticos. Desenvolver modelos de documento (*templates*) para os artefatos sugeridos.

Pode-se também validar a abordagem proposta com alguma abordagem de garantia da qualidade existente, também baseada no PPQA do CMMI-DEV, com o objetivo de estabelecer um paralelo e identificar lacunas e propor aprimoramentos.

Algo ainda a ser considerado pode ser a adequação da proposta com outros modelos de referência, como as normas ISO e o modelo de referência nacional MPS.BR. e outros que visam a garantia da qualidade de software.

## REFERÊNCIAS

---

**BARBACCI**, Mario; Klein, Mark; Longstaff, Thomas; & Weinstock, Charles. *Quality Attributes* (CMU/SEI-95-TR-021 ). Software Engineering Institute, Carnegie Mellon University, 1995. disponível <http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=12433> acessado 08 de março, 2015.

**BARTIÉ**, Alexandre. Garantia da qualidade de software: adquirindo maturidade organizacional / Alexandre Bartié - Rio de Janeiro : Elsevier 2002.

**BASTOS**, A. and Rios, E. and Cristalli, R. and Moreira, T. (2007), Base de Conhecimento em Teste de Software, Martins Editora.

**BEHKAMAL**, Behshid., Mohsen Kahani, Mohammad Kazem Akbari. *Customizing ISO 9126 quality model for evaluation of B2B applications*. Copyright 2008 Elsevier B.V. September 2008, Setembro 2008.

**BOTELLA P.**, X. Burgues, J.P. Carvallo, X. Franch, C. Quer, *Using Quality Models for Assessing COTS Selection*, in: Proceeding of MPEC'04 and ICSE'04, 2004.

**CAROSIA**, J. S. Levantamento da qualidade do processo de software com foco em pequenas organizações / J. S. Carosia. – São José dos Campos: INPE-10424-TDI/921. 2003.

**CESAR**. Centro de Estudos e Sistemas Avançados do Recife - Informática Brasileira em Análise - ano 1, número 2, junho de 1997.

**CHRISSIS**, M. B. and Konrad, M. and Shrum, S. *CMMI for development Guidelines for Process Integration and Product Improvement, Third Edition*. Addison Wesley. 2011.

**CMMI**, *CMMI Product Team*, "CMMI for Development, Version 1.3," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Report CMU/SEI-2010-TR-033, 2010. Disponível em <http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=9661>. Acessado em 13 Março 2015.

**ELBERZHAGER**, F. and Denger, C., *A Comprehensive Planning Framework for Selecting and Customizing Quality Assurance Techniques*, In *IEEE Conference on Software Engineering and Advanced Applications*. 2007. Disponível em <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4301092>, acessado em 18 de março de 2015.

**GONÇALVES**, J. M; Villas Boas, A. Modelo de maturidade da capacidade de software (CMM). Campinas: Fundação CPqD. Versão 1.2. Tradução não oficial do CMU/SEI-93-TR-24-(CMM V1.1), 2001. 60 p.

**ISO/IEC 25010.** *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models*, 2011. Disponível em [http://www.iso.org/iso/catalogue\\_detail.htm?csnumber=35733](http://www.iso.org/iso/catalogue_detail.htm?csnumber=35733), acessado em 14 Março 2015.

**KOCUR**, George. 1.264J *Database, Internet, and Systems Integration Technologies*, Fall 2013. Massachusetts Institute of Technology: MIT OpenCourseWare, disponível em <http://ocw.mit.edu>, acessado 07 de março, 2015.

**MACCALL**, J., P. Richards e G. Walters, “*Factors in Software Quality*”, três volumes, NTIS AD-AO49-014, 015, 055. Novembro 1977.

**MANUVANNAN** Mr. S., *Software process and product quality assurance in IT organizations. International Journal of Computer Engineering. Volume 1, Number 1, May - June (2010)*, pp. 147-157. Junho de 2010.

**MEYER**, B., *Object-oriented Software Construction*, Prentice-Hall, 1988.

**MR-MPS-SW:2012.** MPS.BR - Melhoria de Processo do Software Brasileiro. Guia Geral MPS de Software. Copyright 2012 - SOFTEX, 2012.

**PÁDUA**, Wilson de P.F. Engenharia de Software: fundamentos, métodos e padrões, LTC Livros Técnicos e Científicos, 2003.

**PAULK**, M. C.; Curtis, B; Chississ, M; Weber, C. V. *Capability maturity model version 1.1. Pittsburgh: Software Engineering Institute, Carnegie Mellon University. Technical Report (CMU/SEI-93-TR-024/ESC-TR-93-177)*, Fevereiro 1993. 82 p.

**PRESSMAN**, Roger S. Engenharia de Software: José Carlos Barbosa dos Santos - Sao Paulo : Person Makron Books, 1995.

**WATSON**, Mike. *Managing Smaller Projects: A Practical Approach*. Multi-Media Publications Inc., 2006.